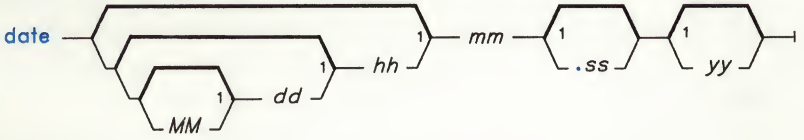
date

Purpose

Displays or sets the date.

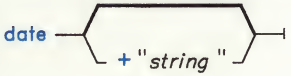
Syntax

Operating With Superuser Authority



OL805105

Operating Without Superuser Authority



[1] Do not put a blank between these items.

OL805357

Description

Warning: Do not change the date while the system is running with more than one user.

If called with no flags or with a flag list that begins with a + (plus sign), the **date** command writes the current date and time to standard output. Otherwise, it sets the current date. Only a user operating with superuser authority can change the date and time. The **NLDATE** variable, if it is defined, controls the ordering of the day and month numbers in the date specifications. The default order is *MMddhhmm.ssyy* where:

- *MM* is the month number
- *dd* is the number of the day in the month
- *hh* is the hour in the day (using a 24-hour clock)
- *mm* is the minute number
- *.ss* is the number of seconds
- *yy* is the last two numbers of the year.

The alternative ordering is *ddMMhhmm.ssyy*.

The current month, day, hour, and year are default values. The system operates in Greenwich Mean Time (GMT). **date** takes care of the conversion to and from local standard and daylight time as specified in the **NLTZ** environment variable.

If you follow **date** with a + (plus sign) and a field descriptor, you can control the output of the command. You must precede each field descriptor with a % (percent sign). The system replaces the field descriptor with the specified value. Enter a literal % as %%. **date** copies any other characters to the output without change. **date** always ends the string with a new-line character. Output fields are fixed size (zero padded if necessary).

Field Descriptors

a Displays the abbreviated day of the week (Sun to Sat or the non-English equivalent).

d Displays the day of month (01 to 31).

D Displays the date as *mm/dd/yy* (the default), or as *dd/mm/yy*. This format is specified by the **NLDATE** environment variable, if defined.

h Displays the abbreviated month (Jan to Dec or the non-English equivalent).

H Displays the hour (00 to 23).

j Displays the day of year (001 to 366).

m Displays the month of year (01 to 12).

M Displays the minute (00 to 59)

n Inserts a new-line character.

r Displays the time in AM/PM notation (or the non-English equivalent).

S Displays the second (00 to 59).

t Inserts a tab character.

T Displays the time as *hh:mm:ss* (the default), or as *mm:hh:ss*. This format is specified by the **NLTIME** environment variable, if defined.

w Displays the day of the week numerically (Sunday = 0).

y Displays the last two numbers of year (00 to 99).

Examples

1. To display current date and time:

```
date
```

2. To set the date and time:

```
date  02171425.45
```

This sets the date and time to 14:25:45 (45 seconds after 2:25 p.m.) February 17 of the current year.

3. To display the date and time in a specified format:

```
date  +"%r  %a  %d  %h  %y  (Julian  Date:  %j)"
```

This displays the date (assume current year is 1984) shown in Example 2 as:

```
02:25:03  PM  Fri  17  Feb  84  (Julian  Date:  048)
```

Files

/dev/kmem

Related Information

See the **time** and **stime** system calls and the **environment** miscellaneous facility in *AIX Operating System Technical Reference.*

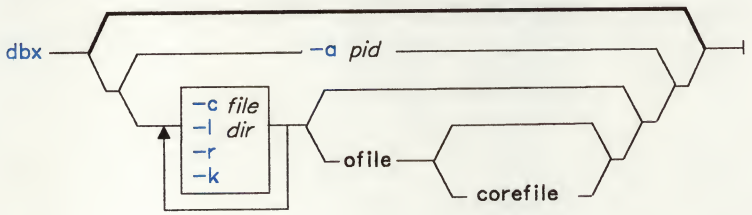
"Overview of International Character Support" in *Managing the AIX Operating System.*

dbx

Purpose

Provides a tool to debug and run programs under AIX.

Syntax



AJ2FL127

Description

The **dbx** command provides a symbolic debugger for C, Pascal, and FORTRAN programs. Use it to do the following:

- Examine object and core files.
- Provide a controlled environment for running a program.
- Set breakpoints at selected statements or run the program one line at a time.
- Debug using symbolic variables and display them in their correct format.

The *ofile* is an object (executable) file produced by a compiler. Use the **-g** (generate symbol table) flag when compiling your program to produce the information **dbx** needs.

Note: If the object file is not compiled with the **-g** flag or if it contains compiler or loader errors, the symbolic capabilities of **dbx** are limited.

When **dbx** is started, it checks for the **.dbxinit** file in the user's current directory. If the file is not found, it checks the user's **$HOME** directory. If **.dbxinit** exists, its subcommands run at the beginning of the debug session. Use an editor to create a **.dbxinit** file.

If the file **core** exists in the current directory or a *corefile* is specified, use the **dbx** debugger to examine the state of the program when it faulted.

When displaying variables and expressions, **dbx** resolves names first using the static scope of the current function. The dynamic scope is used if the name is not defined in the first scope. If static and dynamic searches do not yield a result, an arbitrary symbol is chosen and the system prints the message `[using` *module.variable*`]`. The *module.variable* is the name of an identifier qualified with a block name. Override the name resolution procedure by qualifying an identifier with a block name. Source files are treated as modules named by the file name without the language suffix (such as, the **.f** suffix on a FORTRAN program or the **.c** suffix on a C Language program).

Specify expressions in **dbx** with a subset of C and Pascal (or equivalent Modula-2) syntax. A prefix * or a postfix ^ denotes indirection. Use `[ ]` (square brackets) or `( )` (parentheses) to enclose array subscripts. Use the field reference operator . (period) with pointers and records.

Note: This makes the C operator `->` unnecessary (although it is supported). Specify portions of the array by separating the lower and upper bounds with. (period).

The **dbx** debugger checks types of expressions. Override types of expressions by using *type-name (expression)*. When there is no corresponding named type, use the special construct **&***type-name* to represent a pointer to the named type. Represent a pointer to **enum**, **struct**, or **union tag** with the construct **$$***tag-name*.

The following operators are valid in expressions:

Algebraic	`+, -, *, /` (floating), `div` (integral), `mod, exp` (exponentiation)
Bitwise	`-, |, bitand, xor, ~, <<, >>`
Logical	`or, and, not`
Comparison	`<, >, <=, >=, <> or !=, = or ==`
Other	`sizeof`

Flags

-a *pid* Attaches the debugger to a process that is running. The debugger becomes active as soon as the process wakes up. In order to attach the debugger, you need authority to end a process.

-c *file* Runs the **dbx** commands in the file before reading from standard input.

-I *dir* Includes *dir* in the list of directories searched for source files. The default is to look for source files in the current directory and in the directory where the object file is located. The search path is also set with the **use** subcommand.

-k Maps memory addresses, this is useful for kernel debugging.

-r Runs the object file immediately. If it ends successfully, exit **dbx**. Otherwise, enter the debugger and report the reason for termination.

Note: Unless **-r** is specified, **dbx** prompts the user and waits for a command.

Subcommands

Run and Trace Subcommands

call *proc* (*params*)

Executes the object code associated with the named procedure or function. Use **print** *proc* (*params*) to perform the same function, but with a return code of procedure printed.

catch
catch *signum*
catch *signame*
ignore
ignore *signum*
ignore *signame*

Starts or stops trapping a signal before it is sent to the program. This subcommand is useful when a program being debugged handles signals such as interrupts. A signal is specified by a number or by a name. Signal names are case insensitive. The **SIG** prefix in names is optional. By default all signals are trapped except **SIGHUP**, **SIGCLD**, **SIGALRM**, and **SIGKILL**.

clear *sline*

Removes all stops at a given source line. The *sline* is an integer or a file name string followed by a : (colon) and an integer.

cont
cont *signum*
cont *signame*

Continues execution from the current stopping point until the program finishes or another break point is encountered. If a signal is specified, the process continues as though it received the signal. Otherwise, the process is continued as though it had not been stopped.

delete *num* ...

Removes the traces and stops corresponding to the specified numbers. Use the **status** subcommand to display the numbers associated by **dbx** with a trace or stop.

delete all

Removes all active traces and stops.

detach
detach *signum*
detach *signame*

Continues execution from where it stopped without debugger control. If a signal is specified, the process continues as though it received the signal. Otherwise, the debugger will exit, but the debugged process shall continue.

goto *sline*

Makes the specified source line the next line to be executed.

Note: The source line must be in the same function as the current source line. To override this restriction, set **$vnsafegoto**.

multproc [on]
multproc [off]

Turns on or off multiprocess debugging. The initial value is off. Issue the command without parameters to check the status of multiprocess debugging.

print *proc* (*params*)

Executes the object code associated with the named procedure or function. You use **call** *proc* (*params*) to perform the same function, but with a return code of procedure called.

next [*num*]

Runs the program up to the next source line. This subcommand and the **step** subcommand differ in that if the line contains a call to a procedure or function, **step** will stop at the beginning of that block, and **next** will not. Use *num* to perform a specfic number of **next** commands.

return [*proc*]

Continues until a return to procedure is executed, or until the current procedure returns if none is specified.

run [*args*][< *file*][> *file*] [> > *file*][2> *file*][2> > *file*] [> & *file*][> > & *file*]

rerun [*args*] [< *file*][> *file*] [> > *file*] [2> *file*][2> > *file*] [> & *file*][> > & *file*]

Starts running the object *file*, passing *args* as command line arguments.

< or > or 2>
: Redirects input, output, or standard error, respectively.

>>
: Appends redirected output

2>>
: Appends redirected standard error.

>&
: Redirects both output and standard error to the same file.

>>&
: Appends the redirected output and standard error to the same file.

When **rerun** is used without arguments, the previous argument list is passed.

skip *num*

Continues execution from the current stopping point until *num* + 1 breakpoints are encountered or the program finishes.

status [> *file*]

Displays out the currently active trace and stop commands.

step [*num*]

Runs one source line. Use *num* to execute a specific number of lines.

stop if *cond*
stop at *sline* [**if** *cond*]
stop in *proc* [**if** *cond*]
stop *var* [**in** *proc*] [**if** *cond*]

Stops the program when:

- The condition is true.
- The source line number is reached.
- The procedure (or function) is called.
- The variable is changed.

A condition can be specified for the source line, procedure, or variable stops.

The debugger associates numbers with each **stop** subcommand. Use the **status** subcommand to view these numbers. Use the **delete** or **clear** subcommand to turn stopping off. You use the qualified name to get the actual variable stop.

trace
trace in *proc* [**if** *cond*]
trace *sline* [**if** *cond*]
trace *proc* [**in** *proc*][**if** *cond*]

trace *expr* **at** *sline* [**if** *cond*]

trace *var* [**in** *proc*][**if** *cond*]

Prints the tracing information for the specified procedure (or function), source line, expression, or variable when the program runs. A condition can be specified. The debugger associates numbers with each **trace** subcommand. Use the **status** subcommand to view these numbers. Use the **delete** subcommand to turn tracing off.

watch *var* [**in** *proc*]

Traces changes to a variable in a watch window if invoked under **xdbx**. Otherwise, this is the same as **trace**.

Subcommands for Examining Program Data

assign *var* = *expr* — Assigns the value of the expression to the variable.

case [default]
case [mixed]
case [lower]
case [upper] — Changes the way in which the debugger interprets symbols. The default handling of symbols is based upon the current language. Symbols fold to lowercase unless C is the current language. You use this command if a symbol needs to be interpreted in a way not consistent with the current language. Entering this command with no parameters displays the current case mode.

dump [*proc*] [> *file*] — Displays or puts in a file the names and values of variables in the specified procedure. If the procedure specified is . (period), then all active variables are dumped. The default is the current procedure.

print *expr* [,*expr*...] — Prints out the values of the expressions.

whatis *name* — Displays the declaration of *name* where *name* is a variable, procedure, or function name qualified with a block name.

where [> *file*] — Displays a list of the active procedures and functions.

whereis *identifier* — Displays the full qualification of all the symbols whose name matches the specified identifier. The order in which the symbols print is not significant.

which *identifier* — Displays the full qualification of the given identifier (the outer blocks with which the identifier is associated).

up [*count*]
down [*count*] — Moves the current function, which is used for resolving names, up or down the stack *count* levels. The default is 1.

Subcommands for Accessing Source Files

[*sline-exp* [,sline-exp]]
/*regular expression*[/] — Searches forward in the current source file for the specified pattern.

?*regular expression*[?] — Searches backward in the current source file for the specified pattern.

edit [*file*]
edit *proc* — Invokes an editor with *file* or the current source file if none is specified. If a procedure (or function) *proc* is specified, the editor is invoked on the file that contains it. The default editor

	is **vi**. Override the default by resetting the environment variable **EDITOR** to the name of the desired editor. **Note:** If the procedure has the same name as a file in the same directory, the editor starts the other file, not the file containing the procedure.
file [*file*]	Changes the current source file to *file*, but does not write to the file. If none is specified, displays the name of the current source file.
func [*proc*]	Changes the current function to the specified procedure or function. If none is specified, displays the current function. Changing the current function implicitly changes the current source file to the one containing the function; it also changes the current scope used for name resolution.
list [*proc*]	Lists lines `f-n` to `f+m` where `f` is the first statement in the procedure or function, `n` is a small number, and `m` is the number of lines remaining that fit in the default list window. Use **set $listwindow** = *value* to set or change the number of lines displayed in the list window.
list [*sline-exp*[, *sline-exp*]]	Lists the source lines in the current source file from the first line number to the second inclusive. If no lines are specified, lists the next 10 lines or **$listwindow** lines. An *sline* of $ specifies the current line of execution. An *sline* of `@` specifies the next line to be listed. An *sline-exp* is an *sline* followed by an optional + or - and an integer.
listi [*proc*]	Lists instructions from the specified procedure or function. The number of instructions displayed is controlled by the **$listwindow** value.
listi at *sline*	Lists instructions beginning with the source line specified.
listi [*address* [*address*]]	Lists instructions from the first address to the second address inclusive. If no lines are specified, list the next **$listwindow** instructions.
move *sline*	Changes the next line to be displayed to *sline*. Changes value of `@`.
use *dir* [*dir* ...]	Sets the list of directories to be searched when looking for source files.

Machine Level Subcommands

address,address/[*mode*][>*file*]

address/[*count*][*mode*] [>*file*]

Displays the contents of memory starting at the first address and continuing up to the second address or until count items are printed. If the address is . (period), the address following the one printed most recently is used. The mode specifies how memory is to be printed; if it is omitted, the previous mode specified is used. The initial mode is **X**. The following modes are supported:

b	Prints a byte in octal.
c	Prints a byte as a character.
d	Prints a short word in decimal.
D	Prints a long word in decimal.
f	Prints a single precision real number.
g	Prints a double precision real number.
h	Prints a byte in hexadecimal.
i	Prints the machine instruction.
o	Prints a short word in octal.
O	Prints a long word in octal.
s	Prints a string of characters terminated by a null byte.
x	Prints a short word in hexadecimal.
X	Prints a long word in hexadecimal.

Specify symbolic addresses by preceding the name with an &. Addresses can be expressions made up of other addresses and the operators +, -, and * (indirection). Any expression enclosed in parentheses is interpreted as an address.

cleari *addr*

Remove all the breakpoints at a specified address.

gotoi *addr*

Change program counter address.

registers [>*file*]

Displays the values of all general purpose registers, system control registers, floating point registers, and the current instruction register. General purpose registers are denoted by **$r***n* where *n* is the number of the register. Floating point registers are denoted by **$fr***n*.

stepi [*num*]
nexti [*num*]

Runs a single step as in **step** or **next,** but runs a single instruction rather than source line. If *num* is specified, repeats a single step *num* times.

tracei [*addr*][**if** *cond*]
stopi [*addr*][**if** *cond*]

Traces or sets a stop when the contents of *addr* change.

tracei [*addr*][[**if** *cond*] *addr*][**if** *cond*]

stopi [*var*][**at** *addr*][**if** *cond*]
Turns on tracing or sets a stop at a machine instruction address.

Subcommand Aliases and Variables

alias Displays aliases for subcommands.

alias *name name*
alias *name* "*string*"
alias *name* (*params*) "*string*"
When subcommands are processed, **dbx** checks first to see if the word is an alias for either a subcommand or a string. If it is, **dbx** treats the input as though the corresponding string (with values substituted for any parameters) has been entered.

unalias *name* Removes the alias with the given name.

unset *name* Deletes the debugger variable associated with name.

The set Subcommand

set *var* [= *expr*]
Defines a value (expression) for a debugger variable. The name of the variable cannot conflict with names in the program being debugged. A variable is expanded to the corresponding expression within other commands.

The following variables are selected with **set** and have special meaning:

$dual Turns on both source- and machine-level **dbx** interface with X-Windows.

$expandunions Causes **dbx** to display values of each part of variant records or unions.

$frame Setting this variable to an address causes **dbx** to use the stack frame pointed to by the address for doing stack traces and accessing local variables. This facility is of particular use for kernel debugging.

$hexchars	
$hexin	Causes **dbx** to interpret integers as hexadecimal.
$hexints	
$hexstrings	Causes **dbx** to print out characters, integers, or character pointers respectively in hexadecimal.
$listwindow	Specifies the number of lines to list around a function or when the list command is given without any parameters. Its default value is 10.
$machine	Turns on machine-level **dbx** interface with X-Windows. This variable turns off **$source**.
$mapaddrs	Setting (unsetting) this variable causes **dbx** to start (stop) mapping addresses. This is useful for kernel debugging.
$octin	Causes **dbx** to interpret integers as octal.
$octints	Causes **dbx** to print out integers in octal.
$noargs	Causes **dbx** to omit arguments from commands which walk the stack (where, up, down, dump).
$noflargs	Causes **dbx** to omit display of floating point registers from the **registers** command.
$source	Turns on source-level **dbx** interface with X-Windows.
$unsafeassign	Turns off strict type checking between the two sides of an assign statement. **Note:** Use these variables with great care. They severely limit the usefulness of **dbx** in detecting errors.
$unsafebounds	Turns off subscript checking on arrays.
$unsafecall	Turns off strict type checking for arguments to subroutine or function calls.
$unsafegoto	Turns off **goto** destination checking.

Other Useful Subcommands

help Prints out a synopsis of common **dbx** commands.

prompt "*string*"
Changes the **dbx** prompt to be the same as *string*.

quit Quits **dbx**.

screen Opens a virtual terminal for the **dbx** command interaction. The user continues to operate in the window in which the process originated.

sh *command* Passes the command line to the shell for execution. The **SHELL** environment variable determines which shell is used.

source *file* Reads **dbx** commands from the given file.

Files

a.out	Contains object code; object file.
core	Contains core dump.
.dbxinit	Contains initial commands.

Related Information

The following commands: "**cc**" on page 140 and "**xdbx**" on page 1236.

The **a.out** and **core** files in *AIX Operating System Technical Reference.*

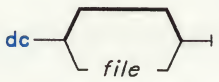
The topic "Debugging Programs" in *AIX Operating System Programming Tools and Interfaces.*

dc

Purpose

Provides an interactive desk calculator for doing arbitrary-precision integer arithmetic.

Syntax



OL805106

Description

The **dc** command is an arbitrary-precision arithmetic calculator. **dc** takes its input from *file* or standard input until it reads an end-of-file character. It writes to standard output. It operates on decimal integers, but you may specify an input base, output base, and a number of fractional digits to be maintained. **dc** is structured overall as a stacking, reverse Polish calculator.

The **bc** command (see page 118) is a preprocessor for **dc**. It provides infix notation and a syntax similar to the C language which implements functions and reasonable control structures for programs.

Subcommands

number	Pushes the specified value onto the stack. A *number* is an unbroken string of the digits 0-9. To specify a negative number, precede it with _ (underscore). A number may contain a decimal point.
+ - / * % ^	Adds (+), subtracts (-), multiplies (*), divides (/), remainders (%), or exponentiates (^) the top two values on the stack. **dc** pops the top two entries off the stack and pushes the result on the stack in their place. **dc** ignores fractional parts of an exponent.
s*x*	Pops the top of the stack and stores it in a register named *x*, where *x* may be any character.
S*x*	Treats *x* as a stack. It pops the top of the main stack and pushes that value onto stack *x*.
l*x*	Pushes the value in register *x* on the stack. The register *x* is not changed. All registers start with zero value.

L*x*	Treats *x* as a stack and pops its top value onto the main stack.
d	Duplicates the top value on the stack.
p	Displays the top value on the stack. The top value remains unchanged. The **p** interprets the top of the stack as an ASCII string, removes it, and displays it.
P	Interprets the top of the stack as a string, removes it, and displays it.
f	Displays all values on the stack.
q	Exits the program. If **dc** is executing a string, it pops the recursion level by two.
Q	Pops the top value on the stack and the string execution level by that value.
x	Treats the top element of the stack as a character string and executes it as a string of **dc** commands.
X	Replaces the number on the top of the stack with its scale factor.
[*string*]	Puts the bracketed *string* onto the top of the stack.
<*x* **>***x* **=***x*	Pops the top two elements of the stack and compares them. Evaluates register *x* as if it obeys the stated relation.
v	Replaces the top element on the stack by its square root. Any existing fractional part of the argument is taken into account, but otherwise the scale factor is ignored.
!	Interprets the rest of the line as a AIX command.
c	Cleans the stack: **dc** pops all values on the stack.
i	Pops the top value on the stack and uses that value as the number radix for further input.
I	Pushes the input base on the top of the stack.
o	Pops the top value on the stack and uses that value as the number radix for further output.
O	Pushes the output base on the top of the stack.
k	Pops the top of the stack, and uses that value as a nonnegative scale factor. The appropriate number of places displays on output and is maintained during multiplication, division, and exponentiation. The interaction of scale factor, input base, and output base is reasonable if all are changed together.

z	Pushes the number of elements in the stack onto the stack.
Z	Replaces the top number in the stack with the number of digits in that number.
?	Gets and runs a line of input.
;:	**bc** uses these characters for array operations.

Examples

1. To use **dc** as a calculator:

```
    You: 1 4 / p
 System: 0
    You: 1 k        [ Keep 1 decimal place  ]s.
         1 4 / p
 System: 0.2
    You: 3 k        [ Keep 3 decimal places ]s.
         1 4 / p
 System: 0.250
    You: 16 63 5 / + p
 System: 28.600
    You: 16 63 5 + / p
 System: 0.235
```

   You may type the comments (enclosed in `[ ]s.`), but they are provided only for your information.

   When you enter **dc** expressions directly from the keyboard, press **Ctrl-D** to end the **bc** session and return to the shell command line.

2. To load and run a **dc** program file:

```
    You: dc prog.dc
         5 lf x p   [  5 factorial ]s.
 System: 120
    You: 10 lf x p  [ 10 factorial ]s.
 System: 3628800
```

   This interprets the **dc** program saved in `prog.dc`, then reads from the work station keyboard.

The `lf x` evaluates the function stored in register `f`, which could be defined in the program file `prog.c` as:

```
[ f: compute the factorial of n ]s.
[    (n = the top of the stack) ]s.

[ If 1>n do b;  If 1<n do r ]s.
   [d 1 >b d 1 <r] sf

[ Return f(n) = 1           ]s.
   [d - 1 +] sb

[ Return f(n) = n * f(n-1)  ]s.
   [d 1 - lf x *] sr
```

You can create **dc** program files with a text editor, or with the **-c** (compile) flag of the **bc** command. When you enter **dc** expressions directly from the keyboard, press **Ctrl-D** to end the **bc** session and return to the shell command line.

Related Information

The following command: "**bc**" on page 97.

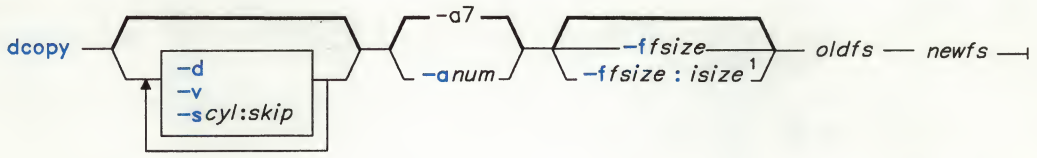
"Overview of International Character Support" in *Managing the AIX Operating System.*

dcopy

Purpose

Copies file systems for the best access time.

Syntax



[1] If not specified, the values from *oldfs* are used.

OL805420

Description

Warning: *oldfs* and *newfs* must not refer to the same minidisk. Doing so will destroy the old file system.

The **dcopy** command copies an existing file system *oldfs* to a new file system *newfs*, appropriately sized to hold the reorganized results. For best results, *oldfs* should be the raw device and *newfs* should be the block device. If *oldfs* or *newfs* is a file system name, **dcopy** uses the corresponding block device given in **/etc/filesystems**. You should run **dcopy** on unmounted file systems (in the case of the root file system, copy to a new minidisk).

If you do not specify any flags, **dcopy** copies files from *oldfs*, compressing directories by removing vacant entries and spacing consecutive blocks in a file by the optimal rotational gap.

The **dcopy** command makes *newfs* identical to *oldfs* and preserves the pack and volume labels. Thus, to compress a file system without moving it, use the **dcopy** command to copy the file to another file system and the **dd** command to copy the file back.

The **dcopy** command catches **INTERRUPT** and **QUIT** signals and reports on its progress. To end **dcopy**, send a **Quit** signal (**Ctrl-V**) and **dcopy** no longer catches **INTERRUPT** or **QUIT**. **dcopy** also attempts to modify its command line arguments so that its progress can be monitored with the **ps** command.

Flags

-a*num*	Places files not accessed in the specified number of days after the free blocks of the destination file system. The default value of *num* is 7. If you do not specify *num*, no files are moved.
-d	Leaves the order of directory entries as is. If you do not specify this flag, **dcopy** moves subdirectories to the beginning of directories.
-f*fsize*[**:***isize*]	Specifies the file system and i-node list sizes (in blocks). If not specified, the value from *oldfs* is used.
-s*cyl*:*skip*	Supplies device information for creating the best organization of blocks in a file, where *cyl* is the number of block per cylinder and *skip* is the number of blocks to skip.
-v	Reports how many files were processed and how big the source and destination free lists are.

Related Information

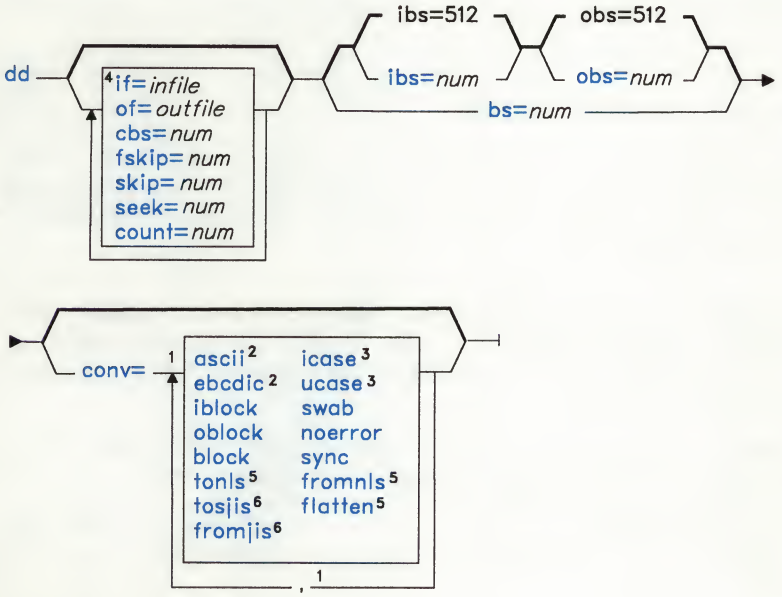
The following commands: "**fsck**, **dfsck**" on page 445, "**mkfs**" on page 658, and "**ps**" on page 786.

dd

Purpose

Converts and copies a file.

Syntax



[1] Do not put a blank between these items.
[2] Use only one of **ascii** and **ebcdic**.
[3] Use only one of **icase** and **ucase**.
[4] *infile* and *outfile* default to standard input and standard output.
[5] Not active when using Japanese Language Support.
[6] Active when using Japanese Language Support.

OL805373

Description

The **dd** command reads the specified *infile* or standard input, does the specified conversions, and copies it to the specified *outfile* or standard output. The input and output block size may be specified to take advantage of raw physical I/O. The terms ***block*** and

record refer to the quantity of data read or written by **dd** in one operation and are not necessarily the same size as a disk block.

Where sizes are specified, a number of bytes is expected. A number may end with **w**, **b**, or **k** to specify multiplication by 2, 512, or 1024 respectively; a pair of numbers can be separated by an **x** to indicate a product.

The conversion requested by **conv=fromnls** translates each extended character in a text file to a printable ASCII escape sequence that uniquely identifies the extended character. The complementary conversion, provided by **conv=tonls**, translates ASCII escape sequences to the corresponding extended character. The conversion requested by **conv=flatten** translates an extended character to the single ASCII character most resembling it in appearance or to a ? (question mark) if no ASCII characters resemble that extended character.

Japanese Language Support Information

The conversion requested by **conv=fromsjis** translates each kanji character in a text file to a printable ASCII escape sequence that uniquely identifies that kanji character. The conversion provided by **conv=tosjis** translates the ASCII escape sequences to the corresponding kanji character.

The character set mappings associated with **conv=ascii** and **conv=ebcdic** are complementary operations, described in the **ebcdic** file in *AIX Operating System Technical Reference*. These attempt to map between ASCII and the subset of EBCDIC that is found on most terminals and keypunches.

The **cbs** specification is used only if the **ascii** or **ebcdic** conversion is specified. For ASCII conversions, **dd** places characters in a conversion buffer of size **cbs**, converts these characters to ASCII, trims trailing blanks and adds new-line characters before sending data specified output. For EBCDIC conversions, it places ASCII characters in the conversion buffer, converts these characters to EBCDIC, adds trailing blanks to create records of size **cbs**.

After it finishes, **dd** reports the number of whole and partial input and output blocks.

Notes:

1. Normally, you need only write access to the output file. However, when the output file is not on a direct access device and you use the **seek** parameter, you also need read access to the file.
2. The **dd** command inserts new-line characters only when converting to ASCII; it pads only when converting to EBCDIC.
3. Use the **backup**, **tar**, or **cpio** commands instead of the **dd** command whenever possible to copy files to tape. These commands are designed for use with tape devices.

4. If you need to use **dd** to copy to a streaming tape and the data is an odd length (not a multiple of 512 bytes), you must use the **conv=sync** option to fill the last record. Streaming tape devices permit only multiples of 512 bytes.

Parameters

if=*infile*	Specifies the input file name; standard input is the default.
of=*outfile*	Specifies the output file name; standard output is the default.
ibs=*num*	Specifies the input block size in bytes; the default is 512.
obs=*num*	Specifies the output block size in bytes; the default is 512.
bs=*num*	Specifies both the input and output block size, superseding **ibs** and **obs**.
cbs=*num*	Specifies the conversion buffer size.
skip=*num*	Skip *num* input records before starting copy.
seek=*num*	Seek to the *num*th record from the beginning of output file before copying.
fskip=*num*	Skip past *num* end-of-file characters before starting copy; this parameter is useful for positioning on multifile magnetic tapes.
count=*num*	Copies only *num* input blocks. The default block size is 512 bytes (see the **ibs** parameter).
conv=*spec*[,*spec* . . .]	Specifies one or more of the following conversions:

ascii	Converts EBCDIC to ASCII.
ebcdic	Converts ASCII to EBCDIC.
tonls	Converts ASCII escape sequences to extended characters.
fromnls	Converts extended characters to ASCII escape sequences.
flatten	Converts extended characters to the ASCII character most resembling it, or to a ? (question mark).

Japanese Language Support Information

tosjis	Converts ASCII escape sequences to kanji characters.
fromsjis	Converts kanji characters to ASCII escape sequences.

iblock

oblock **block**	Minimizes data loss resulting from a read or write error on direct access devices. If you specify **iblock** and an error occurs during a block read (where the block size is 512 or the size specified by **ibs**=*num*), **dd** attempts to reread the data block in smaller size units. If **dd** can determine the sector size of the input device, it reads the bad record one sector at a time. Otherwise, it reads it 512 bytes at a time. The input block size (**ibs**) must be a multiple of this "retry size." This allows you to maximize disk input efficiency while ensuring that data loss associated with a read error is confined to a single sector. The **oblock** conversion works similarly on output. Specifying **block** is same as specifying **iblock,oblock**.
lcase	Makes all alphabetic characters lowercase.
ucase	Makes all alphabetic characters uppercase.
swab	Swaps every pair of bytes.
noerror	Does not stop processing on an error.
sync	Pads every input record to **ibs**.

Example

1. To convert an ASCII text file to EBCDIC:

   ```
   dd if=text.ascii of=text.ebcdic conv=ebcdic
   ```

   This converts `text.ascii` to EBCDIC representation, storing the EBCDIC version in `text.ebcdic`.

   Note: When you specify `conv=ebcdic`, **dd** converts the ASCII ^ (circumflex) character to an unused EBCDIC character (9A hexadecimal), and ASCII ~ (tilde) to EBCDIC ¬ (NOT symbol).

2. To use **dd** as a filter:

   ```
   li -l | dd conv=ucase
   ```

   This displays a long listing of the current directory (`li -l`) in uppercase.

Related Information

The following command: "**cp**" on page 202.

The **ebcdic** and **tape** files in *AIX Operating System Technical Reference.*

"Overview of International Character Support" in *Managing the AIX Operating System.*

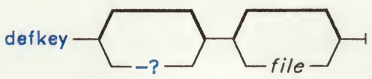
The discussion of Japanese Language Support in *Japanese Language Support User's Guide.*

defkey

Purpose

Defines keyboard key assignments.

Syntax



OL805453

Description

The **defkey** command lets you redefine the keyboard keys on the active virtual terminal. Input to **defkey** comes either interactively from the keyboard or from a redirected file. Key assignments can be a single character, non-spacing characters, or strings.

If you specify a *file* that does not exist, **defkey** creates and opens the file; if *file* exists, **defkey** opens the file. It then displays a menu that prompts you for input. This *file* can then be used as redirected input to **defkey**.

Flags

-? Provides help information.

Examples

1. To redefine a key or keys and create or add to a keyboard definition file:

   ```
   defkey mykeys
   ```

   This creates the file `mykeys` and prompts for input. When **defkey** ends, the keys that you specified will be redefined on the active virtual terminal. You can also use the file `mykeys` to redefine the keyboard on another virtual terminal with the command:

   ```
   defkey  < mykeys
   ```

2. To interactively redefine one or more keyboard keys for the active virtual terminal:

   ```
   defkey
   ```

Related Information

hft and **dispsym** in *AIX Operating System Technical Reference.*

Keyboard Description and Character Reference.

del

Purpose

Deletes files if the request is confirmed.

Syntax

OL805049

Description

The **del** command displays the list of specified *file* names and asks you to confirm your request to delete the group of files. To answer `yes` (delete the files), press the **Enter** key or enter a line beginning with `y`. Any other response specifies `no` (do not delete the files).

Japanese Language Support Information

An affirmative response in Japanese Language Support matches one of the elements in the environment variable **YESSTR**.

The **del** command does not delete directories. See "**rmdir**" on page 838 for information about deleting directories.

Warning: The **del** command ignores file protection, allowing the owner of a file to delete a write-protected file. However, to delete a file, you must have write permission in the directory that the file exists in.

Since pressing the **Enter** key by itself is the same as answering "yes," be careful not to delete files accidentally.

Flag

- Requests confirmation for each specified *file* rather than for the entire group.

Examples

1. To delete a file:

   ```
   del  chap1.bak
   ```

   This displays the message:

   ```
   delete chap1.bak? (y)
   ```

   to ask for confirmation before deleting `chap1.bak`. The `(y)` reminds you to press the **Enter** key or to enter `y` to answer yes.

2. To use **del** with pattern-matching characters:

   ```
   del  *.bak
   ```

   Before passing the command line to **del**, the shell replaces the pattern `*.bak` with the names of all the files in the current directory that end with `.bak`. (This is known as ***file-name expansion***.) **del** asks for confirmation before deleting them all at one time.

3. To interactively select files to be deleted:

   ```
   del  -  *
   ```

   This displays the name of each file in the current directory one at a time, allowing you to select which ones to delete.

Japanese Language Support Information

The allowed affirmative responses are defined in the environment variable **YESSTR**.

Related Information

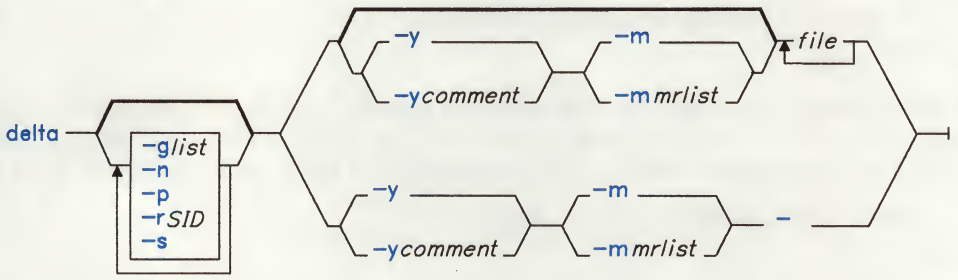
The following commands: "**rmdir**" on page 838 and "**rm**" on page 833.

delta

Purpose

Creates a delta in a Source Code Control System file.

Syntax



OL805056

Description

The **delta** command is used to introduce into the named Source Code Control System (***SCCS***) *file* any changes that were made to the file version retrieved by a **get -e** command.

The **delta** command reads the ***g-files*** that correspond to the specified *file*s (see "SCCS Files" on page 478) and creates a new delta.

If you specify a directory in place of *file*, **delta** performs the requested actions on all SCCS files within that directory (that is, on all files with the **s.** prefix). If you specify a - (minus) in place of *file*, **delta** reads standard input and interprets each line as the name of an SCCS file. When **delta** reads standard input, you must supply the **-y** flag. You must also supply the **-m** flag if the **v** header flag is set. (For more information on header flags, see the discussion in the **admin** command on page 44.) **delta** reads standard input until it reaches END OF FILE (**Ctrl-D**).

If you are not familiar with the delta numbering system, see *AIX Operating System Programming Tools and Interfaces* for more information.

Note: Lines beginning with an **SOH** ASCII character (binary 001) cannot be placed in the SCCS file unless the **SOH** is quoted using a \ (backslash). SOH has special meaning to SCCS and causes an error. See the **sccsfile** file in *AIX Operating System Technical Reference.*

A **get** of many SCCS files, followed by **delta** of those files, should be avoided when the **get** generates a large amount of data. Instead, you should alternate the use of **get** and **delta**.

Flags

-g*list* Specifies a list of *SID*s (deltas) that are to be ignored when the **get** command creates the g-file. After you use this flag, **get** ignores this delta if it is one that it should not include when it builds the g-file.

-m[*mrlist*] If the SCCS file has the **v** header flag set, then a Modification Request (MR) number must be supplied as the reason for creating the new delta.

If you do not specify the **-m** flag, and the **v** header flag is set, **delta** reads MRs from the standard input. If standard input is a work station, **delta** prompts you for the MRs. **delta** continues to take input until it reads END OF FILE (**Ctrl-D**). It always reads MRs before the comments (see the **-y** flag). You can use blanks, tab characters, or both to separate MRs in a list.

If the **v** header flag has a value, it is interpreted as the name of a program that validates the MR numbers. If **delta** returns a nonzero exit value from the MR validation program, **delta** assumes some of the MR numbers were invalid and stops running.

-n Retains the g-file, which is normally removed at completion of **delta** processing.

-p Writes to standard output (in the format of the **diff** command) the SCCS file differences before and after the delta is applied. See "**diff**" on page 320 for an explanation of the format.

-r*SID* Specifies which delta is to be made to the SCCS file. You must use this flag only if two or more outstanding **get -e** commands were done on the same SCCS file by the same person. The *SID* can be either the SID specified on the **get** command line or the SID to be made as reported by the **get** command (see Figure 2 on page 481 for additional information). An error results if the specified SID cannot be uniquely identified, or if a SID must be specified but it is not.

-s Suppresses the information normally written to standard output on normal completion of the **delta** command.

-y[*comment*] Specifies text used to describe the reason for making the delta. A null string is considered a valid *comment*. If your comment line includes special characters or blanks, the line must be enclosed in single or double quotation marks.

If you do not specify **-y**, **delta** reads comments from standard input until it reads a blank line or END OF FILE (**Ctrl-D**). If input is from the keyboard, **delta** prompts for the comments. If the last character of a line is a backslash, it is ignored. Comments must be no longer than 512 characters.

Japanese Language Support Information

Comments can include kanji characters.

Example

To record changes you have made to an SCCS file:

```
delta  s.prog.c
```

This adds a delta to the SCCS file `s.prog.c`, recording the changes made by editing `prog.c`. **delta** then asks you for a comment that summarizes the changes you made. Enter the comment, then press END OF FILE (**Ctrl-D**) or press the **Enter** key twice to indicate that you have finished the comment.

Related Information

The following commands: "**admin**" on page 41, "**bdiff**" on page 102, "**cdc**" on page 152, "**get**" on page 477, "**help**" on page 513, "**prs**" on page 781, and "**rmdel**" on page 837.

The **sccsfile** file in *AIX Operating System Technical Reference.*

The discussion of SCCS in *AIX Operating System Programming Tools and Interfaces.*

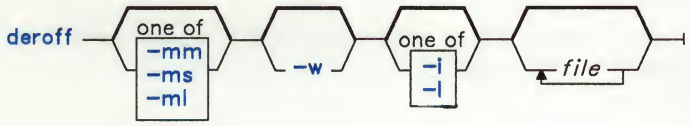
The discussion of Japanese Language Support in *Japanese Language Support User's Guide.*

deroff

Purpose

Removes **nroff, troff, troff, tbl**, and **eqn** constructs from files.

Syntax



OL805181

Description

The **deroff** command reads *file*s (standard input by default), removes all **troff** requests, macro calls, backslash constructs, **eqn** constructs (between **.EQ** and **.EN** lines and between delimiters), and **tbl** descriptions (perhaps replacing them with blanks or blank lines), and writes the remainder of the file to standard output.

The **deroff** command normally follows chains of included files (**.so** and **.nx troff** commands). If a file has already been included, a **.so** naming it is ignored and a **.nx** naming that file ends execution.

Notes:

1. **deroff** is not a complete **troff** interpreter, so it can be confused by subtle constructs. Most errors result in too much rather than too little output.
2. The **-ml** flag does not handle nested lists correctly.

Flags

-i Suppresses the processing of included files.

-l Suppresses the processing of included files whose names begin with **/usr/lib**, such as macro files in **/usr/lib/tmac**.

-mm Ignores MM macros in text so that only running text is output (no text from macro lines is included).

-ml Ignores MM macros in text (**-mm**) and also deletes MM list structures.

-ms Ignores MS macros in text.

-w Makes the output a word list, with one word per line and all other characters deleted. In text, a word is any string that contains at least two letters and is composed of letters, digits, & (ampersands), and ' (apostrophes). In a macro call, a word is a string that begins with at least two letters and contains a total of at least three letters. Delimiters are any characters other than letters, digits, apostrophes, and ampersands. Trailing apostrophes and ampersands are removed from words.

Related Information

The following commands: "**eqn**, **neqn**, **checkeq**" on page 395, "**nroff, troff**" on page 709, "**tbl**" on page 1053, and "**troff**" on page 710.

devices

Purpose

Adds, deletes, changes, and displays device information.

Syntax

OL805306

Description

The **devices** command lets you add, delete, change, or examine information about devices on the system. To use **devices** you must be a member of the system group or have superuser authority.

The **devices** command is an interactive, menu-driven program. For information on how to use it, see *Installing and Customizing the AIX Operating System*. When auditing is enabled for your system, audit records are created. When a device is added or deleted, the audit record is of the type **devices - add** or **devices - del**. When a device is changed, the audit record is of the type **stanza - add** or **stanza - del**.

Files

/etc/filesystems
/etc/predefined
/etc/master
/etc/system
/etc/ports
/etc/qconfig
/tmp/CONFIGREPORT

Related Information

The discussion of **devices** in *Installing and Customizing the AIX Operating System*.

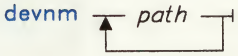
The following command: "**minidisks**" on page 650.

devnm

Purpose

Names a device.

Syntax



OL805114

Description

The **devnm** command reads *path*, identifies the special file associated with the mounted file system where *path* resides, and writes the special file name to standard output. Each *path* must be a full path name.

The most common use of the **devnm** command is by **/etc/rc** to construct a mount table entry for the root device.

Note: This command is for local file systems only.

Examples

1. To identify the device on which a file resides:

   ```
   devnm  /diskette0/bob/textfile
   ```

   This displays the name of the special device file on which `/diskette0/bob/textfile` resides. If a diskette is mounted as **/diskette0**, then **devnm** displays:

   ```
   fd0  /diskette0/bob/textfile
   rfd0  /diskette0/bob/textfile
   ```

   This means that `/diskette0/bob/textfile` resides on the diskette drive **/dev/fd0**.

2. To identify the device on which a file system resides:

   ```
   devnm  /
   ```

   This displays the name of the device on which the root file system (/) resides. The following list appears on the screen:

   ```
   hd0  /
   ```

   This means that / resides on **/dev/hd0**.

Files

/dev	Directory.
/etc/mnttab	Table of mounted devices.

Related Information

The following commands: "**rc**" on page 806 and "**setmnt**" on page 911.

df

Purpose

Reports number of available disk blocks.

Syntax

[1] The default action is to provide information for each file system in **/etc/filesystems** with the attribute `free=true`.

OL805052

Description

The **df** command writes to standard output information about total space and available space on the specified file systems. *filesystem* can be the name of the device on which the file system resides or the directory on which it is mounted. If you do not specify *filesystem*, **df** provides information on all mounted file systems.

Normally, **df** uses free counts maintained in the superblock. Under certain exceptional circumstances, these counts may be in error.

If a file system is being actively modified at the instant **df** is run, the free count may be inaccurate.

Flag

-s This flag is for backwards compatibility only.

Examples

1. To list information about all file systems:

   ```
   df
   ```

   If your system is configured so that the /, **/usr**, **/u**, and **/tmp** directories reside in separate file systems, the output from the **df** command resembles this:

   ```
   Device        Mounted on      total     free  used     ifree  used
   /dev/hd0      /               19368     9976   48%      4714    5%
   /dev/hd1      /usr            24212     4808   80%      5031   19%
   /dev/hd2      /u               9744     9352    4%      1900    4%
   /dev/hd5      /tmp             3868     3856    0%       986    0%
   ```

   Note: On some remote file systems, such as NFS, columns are blank if the server does not provide the information.

2. To list information about the file system on a diskette:

   ```
   df /dev/fd0
   ```

3. To list information about the file system currently mounted as **/diskette0**:

   ```
   df /diskette0
   ```

Related Information

The following command: "**fsck**, **dfsck**" on page 445.

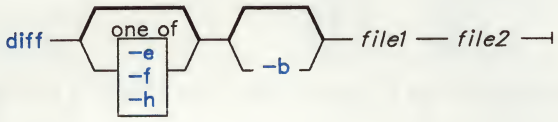
The discussion of **df** in *Managing the AIX Operating System.*

diff

Purpose

Compares text files.

Syntax



OL805046

Description

The **diff** command compares *file1* and *file2* and writes to standard output information about what changes must be made to bring them into agreement. If you specify a - (minus) for *file1* or *file2*, **diff** reads standard input. If *file1* is a directory, then **diff** uses a file in that directory with the name *file2*. If *file2* is a directory, then **diff** uses a file in that directory with the name *file1*.

The normal output contains lines of these forms:

Lines Affected in *file1*	Action	Lines Affected in *file2*
num1	**a**	*num2*[,*num3*]
num1[,*num2*]	**d**	*num3*
num1[,*num2*]	**c**	*num3*[,*num4*]

These lines resemble **ed** subcommands to convert *file1* into *file2*. The numbers before the action letters pertain to *file1*; those after pertain to *file2*. Thus, by exchanging **a** for **d** and reading backward, you can also tell how to convert *file2* into *file1*. As in **ed**, identical pairs (where *num1* = *num2*) are abbreviated as a single number.

Following each of these lines, **diff** displays all lines affected in the first file preceded by a <, then all lines affected in the second file preceded by a >.

Except in rare circumstances, **diff** finds a smallest sufficient set of file differences. An exit value of 0 indicates no differences, 1 indicates differences found, and 2 indicates an error.

Note: Editing scripts produced by the **-e** or **-f** flags cannot create lines consisting of a single . (period).

Flags

-b Ignores trailing spaces and tab characters and considers other strings of blanks to compare as equal.

-e Produces output in a form suitable for use with the **ed** command to convert *file1* to *file2*.

-f Produces output in a form not suitable for use with **ed**, showing the modifications necessary to convert *file1* to *file2* in the reverse order of that produced under the **-e** flag.

-h Performs a faster comparison. This flag only works when the changed sections are short and well separated, but it does work on files of any length. The **-e** and **-f** flags are not available when you use the **-h** flag.

Examples

1. To compare two files:

   ```
   diff chap1.bak chap1
   ```

   This displays the differences between the files `chap1.bak` and `chap1`.

2. To compare two files, ignoring differences in the amount of white space:

   ```
   diff  -b  prog.c.bak  prog.c
   ```

   If two lines differ only in the number of blanks and tabs between words, then **diff** considers them to be the same.

3. To create a file containing commands that **ed** can use to reconstruct one file from another:

   ```
   diff  -e  chap2  chap2.old  >new.to.old.ed
   ```

   This creates a file named `new.to.old.ed` that contains the **ed** commands to change `chap2` back into the version of the text found in `chap2.old`. In most cases, `new.to.old.ed` is a much smaller file than `chap2.old`. You can save disk space by deleting `chap2.old`, and you can reconstruct it at any time by entering:

   ```
   (cat new.to.old.ed ; echo  '1,$p') | ed  -  chap2  >chap2.old
   ```

   The commands in parentheses add `1,$p` to the end of the editing commands sent to **ed**. The `1,$p` causes **ed** to write the file to standard output after editing it. This modified command sequence is then piped to **ed** (`| ed`), and the editor reads it as standard input. The `-` flag causes **ed** not to display the file size and other extra information since it would be mixed with the text of `chap2.old`. See page 931 for details about grouping commands with parentheses.

Files

/tmp/d?????	Temporary files.
/usr/lib/diffh	For the **-h** flag.

Related Information

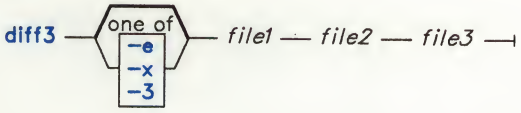
The following commands: "**bdiff**" on page 102 "**cmp**" on page 177, "**comm**" on page 183, "**ed**" on page 371, and "**sdiff**" on page 883.

diff3

Purpose

Compares three files.

Syntax



OL805053

Description

The **diff3** command reads three versions of a file and writes to standard output the ranges of text that differ, flagged with the following codes:

= = = =	All three files differ.
= = = = **1**	*file1* differs.
= = = = **2**	*file2* differs.
= = = = **3**	*file3* differs.

The type of change needed to convert a given range of a given file to match another file is indicated in one of these two ways in the output:

file : *n1* **a**	Text is to be added after line number *n1* in *file*, where *file* is **1**, **2**, or **3**.
file : *n1*[,*n2*] **c**	Text in the range line *n1* to line *n2* is to be changed. If *n1* = *n2*, the range may be abbreviated to *n1*.

The original contents of the range follows immediately after a **c** indication. When the contents of two files are identical, **diff3** does not show the contents of the lower-numbered file, although it shows the location of the identical lines for each.

Notes:

1. Editing scripts produced by the **-e** flag cannot create lines consisting only of a single period (.).
2. The **diff3** command does not work on files longer than 64K bytes.

Flags

-e Creates an edit script for use with the **ed** command to incorporate into *file1* all changes between *file2* and *file3* (that is, the changes that normally would be flagged ==== and ====3).

-x Produces an edit script to incorporate only changes flagged ====.

-3 Produces an edit script to incorporate only changes flagged ====3.

Example

To list the differences among three files:

```
diff3  fruit.a  fruit.b  fruit.c
```

If `fruit.a`, `fruit.b`, and `fruit.c` contain the following data:

fruit.a	fruit.b	fruit.c
banana	apple	grape
grape	banana	grapefruit
kiwi	grapefruit	kiwi
lemon	kiwi	lemon
mango	orange	mango
orange	peach	orange
peach	pear	peach
pare		pear

then the output from **diff3** shows the differences between these files as follows. (The comments on the right do not appear in the output.)

```
====     • All three files are different.
1:1,2c   - Lines 1 and 2 of the first file, fruit.a
  banana
  grape
2:1,3c   - Lines 1 through 3 of fruit.b
  apple
  banana
  grapefruit
3:1,2c   - Lines 1 and 2 of fruit.c
  grape
  grapefruit
====2    • The second file, fruit.b, is different.
```

```
1:4,5c    - Lines 4 and 5 are the same in fruit.a and fruit.c.
2:4a      - To make fruit.b look the same, add text after line 4.
3:4,5c
  lemon
  mango
====1     • The first file, fruit.a, is different.
1:8c
  pare
2:7c      - Line 7 of fruit.b and line 8 of fruit.c are the same.
3:8c
  pear
```

Files

/tmp/d3*
/usr/lib/diff3prog

Related Information

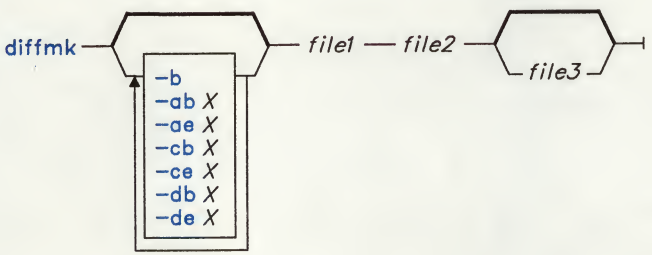
The following command: "**diff**" on page 320.

diffmk

Purpose

Marks differences between files.

Syntax



OL805057

Description

The **diffmk** command compares *file1* and *file2* and creates a third file that includes ***change mark commands*** for the **nroff** and **troff** commands. *file1* and *file2* are the old and new versions of the file. **diffmk** writes the newly created file to *file3*, if specified, or to standard output. This file contains the lines of *file2* with formatter change mark (**.mc**) requests inserted as appropriate. When *file3* is formatted, the changed or inserted text is marked by a | (vertical bar) at the right margin of each line. An * (asterisk) in the margin indicates that a line was deleted.

If the environment parameter DIFFMARK is defined, it names a command string that **diffmk** uses to compare the files. (Normally, **diffmk** uses the **diff** command.) For example, you might set DIFFMARK to `diff -h` in order to better handle extremely large files.

Flags

-ab*X* Uses *X* to mark where added lines begin.

-ae*X* Uses *X* to mark where added lines end.

-b Ignores differences that are only changes in tabs or spaces on a line.

-cb*X* Uses *X* to mark where changed lines begin.

-ce*X* Uses *X* to mark where changed lines end.

-db*X* Uses *X* to mark where deleted lines begin.

-de*X* Uses *X* to mark where deleted lines end.

Examples

1. To mark the differences between two versions of a text file:

```
diffmk  chap1.old  chap1  > chap1.nroff
```

This produces a copy of `chap1` containing **nroff/troff** change mark commands to identify text that has been added to, changed in, or deleted from `chap1.old`. This copy is saved in the file `chap1.nroff`.

2. To mark differences with non-**nroff/troff** messages:

```
diffmk -ab'>>New:' -ae'<<End New' chap1.old  chap1  >chap1.nroff
```

This causes **diffmk** to write `>>New:` on the line before a section of new lines that have been added to `chap1` and to write `<<End New` on the line following the added lines. Changes and deletions still generate **nroff/troff** commands to put a ! or * in the margin.

3. To use different **nroff/troff** marking commands and ignore changes in white space:

```
diffmk  -b  -cb'.mc  %'  chap1.old  chap1  > chap1.nroff
```

This imbeds commands that mark changes with %, additions with !, and deletions with *. It does not mark changes that only involve a different number of spaces or tabs between words (`-b`).

Related Information

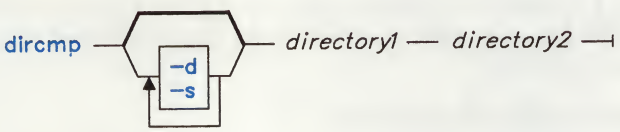
The following commands: "**diff**" on page 320, "**nroff, troff**" on page 709, and "**troff**" on page 710.

dircmp

Purpose

Compares two directories and the contents of their common files.

Syntax



OL805004

Description

The **dircmp** command reads *directory1* and *directory2* and writes information about their contents to standard output. First, **dircmp** compares the file names in each directory. When the same file name appears in both, **dircmp** compares the contents of both files.

In the output, **dircmp** lists the files unique to each directory. It then lists the files with identical names in both directories, but with different contents. With no flag, it also lists files that have identical contents as well as identical names in both directories.

Flags

-d Displays for each common file name both versions of the differing file lines. The display format is the same as that of "**diff**" on page 320.

-s Does not list the names of identical files.

Examples

1. To summarize the differences between the files in two directories:

   ```
   dircmp  proj.ver1  proj.ver2
   ```

   This displays a summary of the differences between the directories `proj.ver1` and `proj.ver2`. The summary lists separately the files found only in one directory or the other, and those found in both. If a file is found in both directories, **dircmp** notes whether or not the two copies are identical.

2. To show the details of the differences between files:

```
dircmp  -d  -s  proj.ver1  proj.ver2
```

The **-s** flag suppresses information about identical files. The **-d** flag displays a **diff** listing for each of the differing files found in both directories.

Related Information

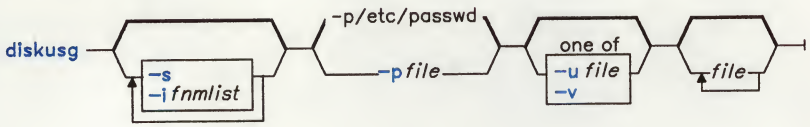
The following commands: "**cmp**" on page 177 and "**diff**" on page 320.

diskusg

Purpose

Generates disk accounting data by user ID.

Syntax



OL805402

Description

The **diskusg** command generates intermediate disk accounting information from data in *files* or from standard input if you do not specify any files. **diskusg** writes lines to standard output, one per user, in the following format:

uid login #blocks

where:

uid	Is the numerical user ID of the user
login	Is the login name of the user; and
#blocks	Is the total number of disk blocks allocated to this user.

The **diskusg** command normally reads only the i-nodes of file systems for disk accounting. In this case, *files* are the special file names of these devices.

Note: This command is for local devices only.

Japanese Language Support Information

This command has not been modified to support Japanese characters.

Flags

-i *fnmlist* Ignores the data on those file systems with a file system name in *fnmlist*. *fnmlist* is a list of file system names separated by commas or enclosed within quotation marks. **diskusg** compares each name in this list with the file system name stored in the volume ID.

-p *file* Uses *file* as the name of the password file to generate login names. **/etc/passwd** is used by default.

-s Combines all lines for a single user into a single line. (The input date is already in **diskusg** output format.)

-u *file* Writes records to *file* of files that are charged to no one. Records consist of the special file name, the i-node number, and the user ID.

-v Writes a list to standard error of all files that are charged to no one.

The output of **diskusg** is normally the input to **acctdisk**, which generates total accounting records that can be merged with other accounting records. **diskusg** is normally run in **dodisk** (see "**acct/***" on page 13).

Examples

The following will generate daily disk accounting information:

```
for  i  in  /dev/hd0  /dev/hd1  /dev/hd2  /dev/hd3
do
     diskusg  $i  > dtmp.'basename $i'  &
done
wait
diskusg  -s  dtmp.*  ¦ sort  +0n  +1  ¦  acctdisk  > dacct
```

Files

/etc/passwd Used for user ID to login name conversions.

Related Information

The following commands: "**acct/***" on page 13, "**acctcms**" on page 18, "**acctcom**" on page 20, "**acctcon**" on page 24, "**acctmerg**" on page 28, "**acctprc**" on page 30, "**fwtmp**" on page 457, and "**runacct**" on page 848.

The **acct** system call and the **acct** and **utmp** files in *AIX Operating System Technical Reference*.

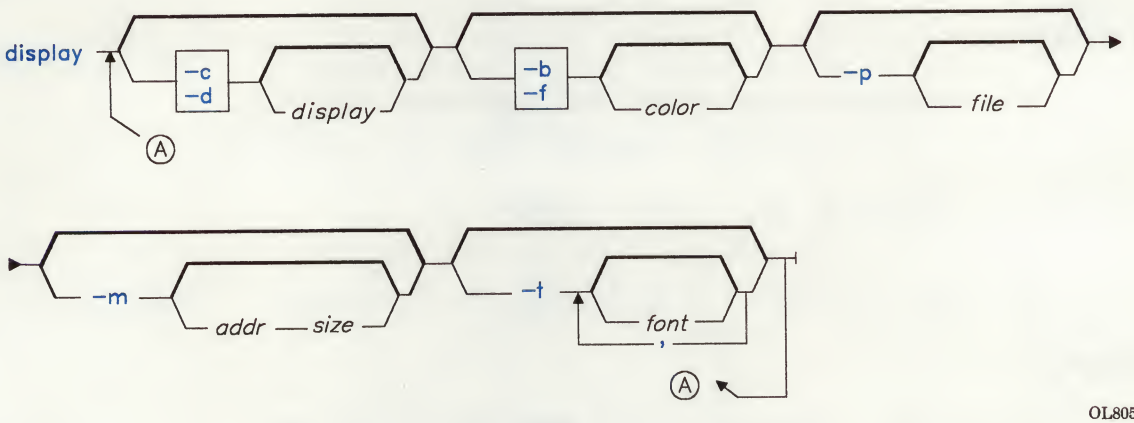
The discussion of accounting in *Managing the AIX Operating System*.

display

Purpose

Selects the physical display that an existing or new virtual terminal uses and sets colors and fonts.

Syntax



OL805442

Description

The **display** command changes the physical display assigned to the current virtual terminal or assigns a default display to be used when you open a virtual terminal. It also sets the foreground and background colors, the active color palette, and the active and alternate fonts on the current display. The *display* parameter can be one of the following names:

pcmono	PC Monochrome Adapter and Display
egamono	Enhanced Graphics Adapter and PC Monochrome Display
egacol	Enhanced Graphics Adapter and Display
advmono	Advanced Monochrome Graphics Adapter and Display
advcol	Advanced Color Graphics Adapter and Display
extmono	Extended Monochrome Graphics Adapter and Display
megapel	IBM Megapel Display Adapter and IBM 5081 Display Models 16 and 19.

You can request only those displays that are actually installed on the system. If you have more than four, only four will be displayed on the **-c** and **-d** menus. Before **display** makes any changes, it checks all arguments for errors and, if it encounters one, displays a list of valid arguments and exits.

Note: You must insure that the **TERM** shell variable contains the proper value for whatever the current display is. See "**termdef**" on page 1062 and the **terminfo** file in *AIX Operating System Technical Reference* for a list of these values.

Flags

-b [*color*]
Selects the background color. The *color* parameter is an integer from 1 to 8 for the Enhanced Graphics Display and from 1 to 16 for other color graphics displays. These values correspond to the first eight or sixteen entries in the active color palette (see the **-p** flag). For example, `-b 5` selects the fifth entry. If you do not specify a number, **display** lists the palette of active background colors and prompts you to select a number for the new background color.

-c [*display*]
Changes the display used by the current virtual terminal.

If you do not specify a *display*, you are given a menu of available options. This menu consists of a numbered list of display names and descriptions. The display number reflects the number of physical displays installed and their relative positions in the Real Screen Table. The current default is always display number 1 in this list. Changing the default alters the display number associated with each physical display. If the virtual terminal does not know the display/adapter combination, the Name column will contain the words `Unknown Display` or `?????`. A prompt at the bottom of the display list asks you to enter the new display number for the current or default display setting. Whenever you change the current display, the screen of that display clears.

-d [*display*]
Changes the default display used when a virtual terminal is opened. If you do not specify a *display*, you are given a menu of available options (see the **-c** flag).

-f [*color*]
Selects the foreground color. The *color* parameter is an integer from 1 to 16. These values correspond to the first sixteen entries in the active color palette (see the **-p** flag). If you do not specify a color number, **display** lists the palette of active foreground colors and prompts you to select a number for the new foreground color.

-m [*addr size*]
Changes the DMA pinned page at the specified starting address to *size* 256K blocks. If you do not specify an address and a size, the current starting address and size is displayed.

-p [*file*] Changes the active color palette. The optional *file* parameter is the full path name to a file that contains a list of colors for the current display, one color per line, where each color is the decimal representation of the 32-bit color value. The color palette file can also contain blank lines and comment lines (a comment line must begin with a * character in column one). Each supported display has a corresponding color file which contains its default active color palette. The name of this file is **/etc/vtm/pal.***name* where *name* is the display name described on page 332. This is the default value for the *file* parameter.

-t [*font*[,*font*] . . .] Selects the primary and active alternate fonts for the current virtual terminal on the current display. The first *font* named in the optional list following **-t** will be the primary font. The remaining fonts will be alternates, in the order listed, for the active font table. If you do not specify eight font IDs, the first font will be used to fill out in the remaining entries in the active font table.

Notes:

1. All of the fonts in the list must be of the same size.
2. Some applications that use the **terminfo** file expect the italic font to be the first alternate and the bold font to be the second alternate fonts (see the **terminfo** file in *AIX Operating System Technical Reference* for more information).

If you do not specify any fonts, all of the fonts available for the current display will be listed, and you will be prompted first for the desired primary font ID and then for alternate font IDs until you enter F. As you enter alternate fonts, the **display** command checks that they are the same size as the new primary font. If you enter fewer than eight fonts, the primary font will be repeated in the remaining entries of the active font table.

You can specify combinations of the same flags on a single command line. **display** processes **-c** and **-d** flags first. If you specify **-c**, you will see the message `Changing to current display...`, and the current display will be changed. Any menu interface for the color or font parameters will be displayed there. A **-p** flag will be processed next. The screen will be immediately redrawn with the colors from the new color palette. Then any foreground, background, or font flags will be processed.

Examples

1. To change the current virtual terminal display:

```
display  -c  egamono
```

This changes the display to the Enhanced Graphics Adapter and PC Monochrome Display.

2. To make the Advanced Color Graphics Display the default virtual terminal display:

```
display -d advcol
```

3. To change both the current and the default displays:

```
display -c pcmono -d egacol
```

This makes the PC Monochrome Adapter and Display the current display and makes the Enhanced Graphics Adapter and Display the default display.

4. To change the active color palette for the current display:

```
display -p /u/new/palette
```

Related Information

The following commands: "**open**" on page 728 and "**termdef**" on page 1062.

The **terminfo** file in *AIX Operating System Technical Reference.*

"Using Display Station Features" in *IBM RT Using the AIX Operating System* and "Managing Display Station Features" in *IBM RT Managing the AIX Operating System.*

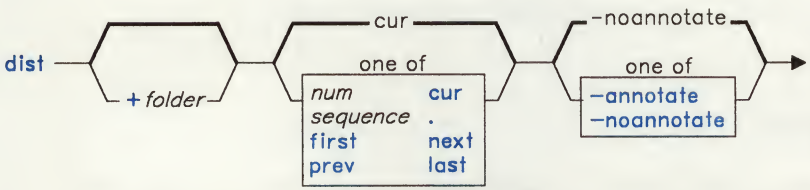
The default color palettes in *Virtual Resource Manager Technical Reference.*

dist

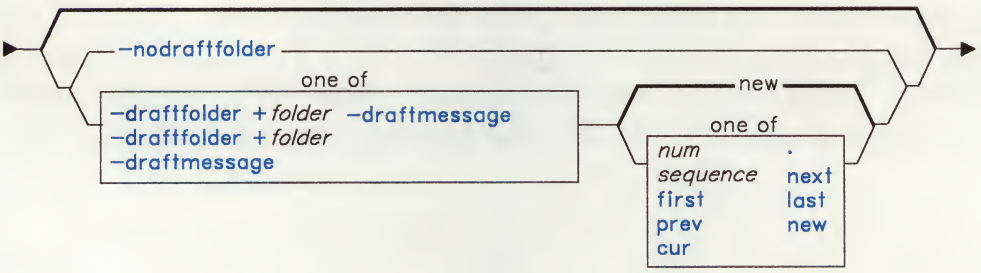
Purpose

Redistributes a message to additional addresses.

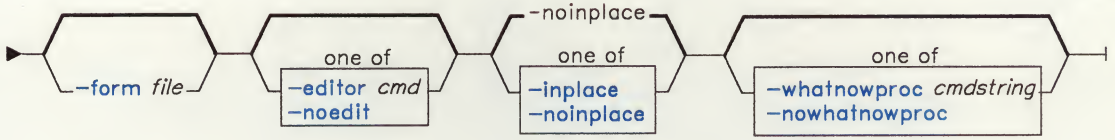
Syntax

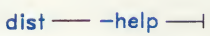


AJ2FL242



AJ2FL157





AJ2FL243

Description

The **dist** command is used to redistribute messages to a new list of addresses. **dist** is part of the MH (Message Handling) package and can be used with other MH and AIX commands.

By default, **dist** copies a message form to a new draft message and invokes an editor. You can then fill in the message header fields **Resent-To:** and **Subject:** and fill in or delete the other header fields (such as **Resent-cc:** and **Resent-Bcc:**). Since the body of the message will be the message you are redistributing, do not fill in the body. **dist** does not automatically display the body of the message. When you exit the editor, the **dist** command invokes the MH command **whatnow**. You can press **Enter** to see a list of the available **whatnow** subcommands. These subcommands enable you to continue editing the message header, list the message header, direct the disposition of the message, or end the processing of the **dist** command. "**whatnow**" on page 1215 describes the subcommands.

When you send the draft message, the recipients are sent the headers and body of the original message appended to the new message. **dist** does not automatically store a copy of the original message with the new draft message. The draft message you create using the **dist** command consists of header fields only.

You can specify the message that you want to distribute by using the *+folder msg* flag. If you do not specify a message, **dist** redistributes the current message.

You can specify the format, of the message header by using the **-form** flag. If you do not specify this flag, **dist** uses your default message format located in the file *user_mh_directory*/**distcomps**. If this file does not exist, **dist** uses the system default message format located in **/usr/lib/mh/distcomps**. **dist** prepends the form to the message being redistributed.

Note: The line of dashes or a blank line must be left between the header and the body of the message for the message to be identified when it is sent.

Flags

-annotate Annotates the message being redistributed with the lines:

Resent: *date*
Resent: *addrs*

The annotation appears in the original draft message so that you can maintain a complete list of recipients with the original message. If you do not actually redistribute the message using the immediate **dist** command, the **-annotate** flag may fail to provide annotation. The **-inplace** flag forces annotation to be done in place.

-draftfolder +*folder*
Places the draft message in the specified folder. If you do not specify this flag, **dist** selects a default draft folder according to the information supplied in the MH profiles. You can define a default draft folder in **$HOME/.mh_profile**. If **-draftfolder** +*folder* is followed by *msg*, *msg* represents the **-draftmessage** attribute.

-draftmessage *msg*
Specifies the draft message. You can specify one of the following message references as *msg*:

num	*sequence*	**first**
prev	**cur**	**.**
next	**last**	**new**

The default draft message is **new**. If you specify a draft message, that message becomes the current message.

-editor *cmd*
Specifies that *cmd* is the initial editor for preparing the message for distribution. If you do not specify this flag, **dist** selects a default editor or suppresses the initial edit, according to the information supplied in the MH profiles. You can define a default initial editor in **$HOME/.mh_profile**.

+*folder msg*
Redistributes the specified message in the specified folder. You can specify one of the following message references as *msg*:

num	*sequence*	**first**
prev	**cur**	**.**
next	**last**	

The default message is the current message in the current folder. If you specify a folder, that folder becomes the current folder.

-form *file*
Prepends the form contained in the specified file to the message being resent. **dist** treats each line in *file* as a format string.

-help
Displays help information for the command.

-inplace
Forces annotation to be done in place in order to preserve links to the annotated message.

-noannotate
Does not annotate the message. This flag is the default.

-nodraftfolder
Places the draft in the file *user_mh_directory*/**draft**.

-noedit
Suppresses the initial edit.

-noinplace
Does not perform annotation in place. This flag is the default.

-nowhatnowproc Does not invoke a program that guides you through the distribution tasks. The **-nowhatnowproc** flag also prevents any edit from occurring.

-whatnowproc *cmdstring* Invokes *cmdstring* as the program to guide you through the distribution tasks. See "**whatnow**" on page 1215 for information about the default **whatnow** program and its subcommands.

Note: If you specify `whatnow` for *cmdstring*, **dist** invokes an internal **whatnow** procedure rather than a program with the file name **whatnow**.

Profile Entries

Current-Folder:	Sets your default current folder.
Draft-Folder:	Sets your default folder for drafts.
Editor:	Sets your default initial editor.
fileproc:	Specifies the program used to refile messages.
Path:	Specifies your *user_mh_directory*.
whatnowproc:	Specifies the program used to prompt `What now?` questions.

Files

/usr/lib/mh/distcomps	The system default message skeleton
user_mh_directory/distcomps	The user's default message skeleton. (If it exists, it overrides the system default message skeleton.)
$HOME/.mh_profile	The MH user profile.
user_mh_directory/draft	The draft file.

Related Information

Other MH commands: "**ali**" on page 48, "**anno**" on page 50, "**comp**" on page 185, "**forw**" on page 438, "**prompter**" on page 778, "**refile**" on page 817, "**repl**" on page 821, "**send**" on page 893, "**whatnow**" on page 1215.

The **mh-alias**, **mh-format**, **mh-mail**, and **mh-profile** files in *AIX Operating System Technical Reference.*

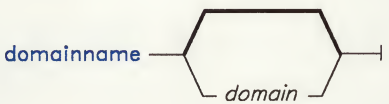
"Overview of the Message Handling Package" in *Managing the AIX Operating System.*

domainname

Purpose

Sets or displays the name of the current Yellow Pages (YP) domain.

Syntax



OL805481

Description

The **domainname** command displays the name of the current YP domain. If you have superuser authority, you can also use this command to set the name of the domain.

A ***domain*** is a group of host machines in the network. The name of the domain typically is set in **/etc/rc.nfs** file.

Japanese Language Support Information

If Japanese Language Support is installed on your system, this command is not available.

File

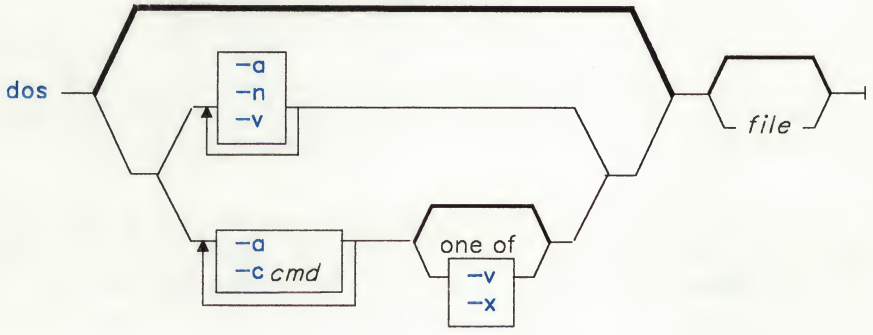
/etc/rc.nfs The NFS startup shell script.

dos

Purpose

Starts **shell**.

Syntax



OL805330

Description

The **dos** command starts a DOS emulation environment. It interprets DOS commands and runs programs that can use the routines that simulate DOS run-time behavior. (For more information on these routines and this environment, see *DOS Services Reference* and *Installing and Customizing the AIX Operating System*.)

When you enter **dos**, a DOS environment file is created from the process environment. (For details on how this is done, see **dosinit** in *AIX Operating System Technical Reference*.) Upon invocation, **dos** sets the current drive to **A** or the first valid drive. The environment variable **DOSDISK** can be set to define the default current drive (**B**, **C**, **D**, and so on).

The *file* parameter specifies a **dos** batch file to be run. *file* must have the extension **.bat** or **.BAT**.

If the current DOS Services directory contains the batch file **autoexec.bat** or **AUTOEXEC.BAT**, then DOS Services initially reads and runs commands from this file.

DOS commands are either built-in (to the **dos** command itself), or they are external. External commands reside in the **/usr/dos/bin** directory. Normally, the search order for commands that you enter is as follows:

- The directory **/usr/dos/bin**
- The working directory
- Each directory in the **dos** path.

When you enter a command, **dos** searches each directory for a file with a name composed of the command name and either the extension **.BAT**, the extension **.bat**, or no extension. If the file has the extension **.BAT** or **.bat**, it runs as a batch file. Otherwise, it runs as an AIX program. If it is an AIX program, it can be either a compiled program or a shell file. In either case you must have execute access to it.

The **dos** command supports two types of file systems: AIX file systems and DOS file systems. Each **dos** minidisk can contain either an AIX-formatted file system or a DOS-formatted file system. However, diskette drives (such as **/dev/fd0**) may contain only DOS-formatted file systems, unless the device is mounted as an AIX file system before you invoke **dos**.

Warning: Only one user or process at a time can access a **dos** file system. If a **dos** file system resides on a minidisk, two or more users may attempt to access the minidisk at the same time. Because **dos** has no way to warn you that another process is using a minidisk, you should allocate minidisks containing **dos** file systems on a per-user basis.

If a coprocessor on the system accesses a **dos**-formatted minidisk at the same time as an RT process, there is no conflict because only the first process has read/write privileges. Subsequent opens at the device level are limited to read-only access.

There are different restrictions for file names on DOS drives and AIX drives. For DOS Services drives:

- File names cannot be longer than 12 characters.
- The name is always stored in uppercase.
- All files in the directory must have unique names.
- There can be only one period in a file name.

For AIX file systems:

- File names cannot be longer than 14 characters.
- Names may contain either uppercase or lowercase letters.
- Two files in the same directory can have the same name if the letter case is different.
- There can be more than one period in a file name.
- All files in the directory must have unique names.

On AIX drives, file names that begin with a period specify hidden files. On DOS Services drives, hidden files have a bit set in the attribute byte of the file directory.

There are differences between AIX and DOS Services file formats. AIX ASCII files and DOS Services ASCII files are similar and can be converted from one format to the other. Two new commands, **FILETYPE** and **CONVERT**, are available for detecting and changing a file format.

DOS Services Commands and Programs

There are several differences between the set of supported DOS Services commands and DOS commands.

Unsupported DOS Commands and Programs

You can use all of the standard DOS commands except **BREAK**, **CTTY**, **EDLIN**, **EXE2BIN**, **GRAPHICS**, and **SYS**.

Modified DOS Commands

The following DOS Services commands behave differently than the corresponding standard DOS commands:

backup The **/M** parameter is not valid for DOS Services file systems.

chdir Unlike DOS, DOS Services may not allow you to change to the highest directory in the file system.

date This command lets only the superuser change the date.

dir Does not list file-name extensions in a separate column when executed on an AIX drive.

format The **/B** is not supported. Two additional flags, **/U** and **/H** are supported. Use the **/U** flag to format a AIX diskette. Use the **/H** flag to format a fixed disk to contain DOS Services file systems in a single partition.

Note: The **format** command makes use of the **mksf** command, which in turn uses the **/etc/filesystems** file. If you modify this file, it will affect the **format** command.

label On an AIX-formatted drive, the label is written to a file called **LABEL.VOL**. Reading a label is accomplished by reading this file. Changing a label modifies the contents of this file.

Note: The command **del *.*** deletes the volume label.

mode Only option 3 (for an asynchronous communications adapter) is supported.

print The DOS Services version does not ask you which device to store the print queue on. This information is set up in your user profile.

The **/B**, **D**, **M**, **/S**, **/Q**, and **/U** configuration flags are not supported.

set A /U flag lets you display the AIX environment as it is inherited by the **dos** command. You can change the environment variables internal to **dos**. When you exit from **dos**, the environment variables remain unchanged.

time Allows only the superuser to change the time.

Additional Commands

In addition to DOS commands, the following commands are available:

COMMAND The new flags which have been added to **dos** also apply to this command.

CONVERT Converts a DOS format ASCII file to an AIX format ASCII file or an AIX format ASCII file to a DOS format ASCII file.

ed Starts the line editor.

EXIT Ends DOS Services. You can also use END OF FILE (**Ctrl-D**).

FILETYPE Attempts to determine the format (AIX or DOS) and contents of the specified file.

shutdown Provides for an orderly exit from the system.

Flags

-a Does not run the **AUTOEXEC.BAT** file.

-c *cmd* Runs the specified command.

-n Reads commands but does not run them.

-v Displays the commands and their flags as they are read.

-x Displays the commands and their flags as they are run.

Files

/usr/dos/bin/*	DOS Services external commands.
AUTOEXEC.BAT	Batch file that can run commands automatically.
autoexec.bat	Batch file that can run commands automatically.

Related Information

The following commands: "**dosdel**" on page 345, "**dosread**" on page 348 and "**doswrite**" on page 350.

The **dosinit** subroutine in *AIX Operating System Technical Reference.*

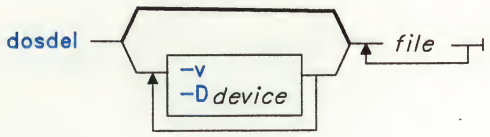
The discussion of **dos** in *Using DOS Services* and *DOS Services Reference.*

dosdel

Purpose

Deletes DOS files.

Syntax



OL805108

Description

The **dosdel** command deletes the DOS file specified by *file*. Use the **-v** flag to obtain format information about the disk.

File-naming conventions are those of DOS, with one exception. **doswrite** replaces the \ (backslash) character used to separate components of a DOS path name with the / (slash) because the backslash can have special meaning to AIX. **dosdel** converts lowercase characters in the *file1* name to uppercase before it checks the disk. Because all file names are assumed to be full (not relative) path names, you need not add the initial / (slash).

Flags

-D *device* Specifies a device or file system to use as the DOS disk. If you do not specify this flag the default device is **/dev/fd0**.

-v Writes format information about the disk. Use primarily to verify the identify of a disk or file system as a DOS disk.

Related Information

The following commands: "**dos**" on page 341, "**dosdir**" on page 346, "**dosread**" on page 348, and "**doswrite**" on page 350.

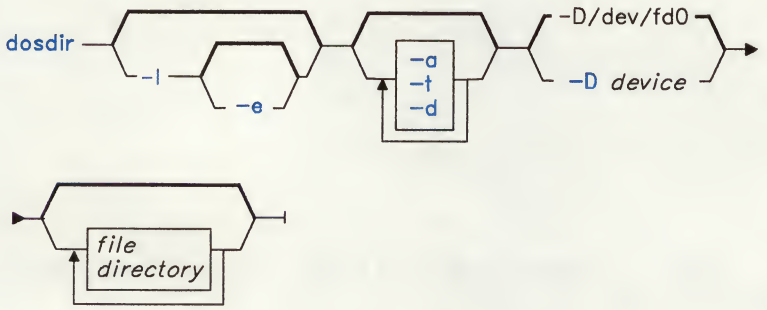
The **pcdos** subroutine in *AIX Operating System Technical Reference.*

dosdir

Purpose

Lists the directory for DOS files.

Syntax



OL805358

Description

The **dosdir** command displays information about the specified DOS file or directory (the current directory by default). If you specify a directory without also specifying the **-d** flag, **dosdir** displays information about the files in that directory.

File-naming conventions are those of DOS, with one exception. **dosdir** replaces the \ (backslash) character used to separate components of a DOS path name with a / (slash) because the backslash can have special meaning to the AIX Operating System. **dosdir** converts lowercase characters in the file or directory name to uppercase before it checks the disk. Because all file names are assumed to be full (not relative) path names, you need not add the initial / (slash).

Flags

-a Writes information about all files. This includes hidden and system files as well as the . (dot) and .. (dot dot) files.

-d Treats *file* as a file, even if it is a directory. If a directory is specified, information about the directory is listed rather than information about the files it contains.

-D [*device*] Specifies a device or file system to use as the DOS disk. If you do not specify this flag the default device is **/dev/fd0**.

-e Uses the **-l** flag to write the list of clusters allocated to the file.

-l Produces a long list that includes the creation date, size in bytes, and attributes. The size of a subdirectory is specified as 0 bytes. The attributes have the following meanings:

A Archive - the file has not been backed up since it was last modified.
D Directory - the file is a subdirectory, and is not included in the normal DOS directory search.
H Hidden - the file is not included in the normal DOS directory search.
R Read-only - the file cannot be modified.
S System - the file is a system file, and is not included in the normal DOS directory search.

-t Lists the entire directory tree starting at the named directory.

-v Writes information about the format of the disk.

Related Information

The following commands: "**dosdel**" on page 345, "**dosread**" on page 348, and "**doswrite**" on page 350.

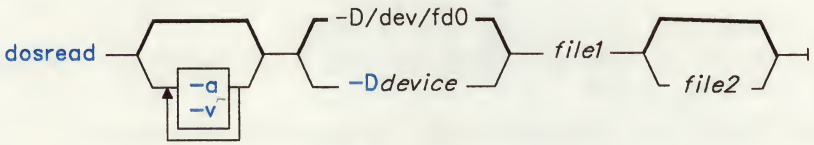
The **pcdos** subroutine in *AIX Operating System Technical Reference.*

dosread

Purpose

Copies a DOS file.

Syntax



OL805111

Description

The **dosread** command copies the specified DOS *file1* to standard output or to the specified AIX *file2* (by default the root directory). Unless otherwise specified, **dosread** copies as many bytes as are specified in the directory entry for *file1*. This means, in particular, that copying directories does not work, since directories by convention have a record size of 0.

File-naming conventions are those of DOS, with one exception. **dosread** replaces the \ (backslash) character used to separate components of a DOS path name with a / (slash) because the backslash can have special meaning to the AIX Operating System. **dosread** converts lowercase characters in the *file1* name to uppercase before it checks the disk. Because all file names are assumed to be full (not relative) path names, you need not add the initial / (slash).

Notes:

1. Wild card characters (* and ?) are not treated in a special way by this command (although they are by the shell). If, for example, you do not specify a file-name extension, the file name is matched as if you had specified a blank extension.
2. This command must be named **dosread**.

Flags

-a	Replaces the sequence **CR-LF** (carriage return-line feed) with **NL** (new-line character) and interprets a **Ctrl-Z** (ASCII SUB) as the end-of-file character.
-D *device*	Specifies the name of the DOS device or file system. The default *device* is **/dev/fd0**. This device must have the DOS-disk format.
-v	Writes information to the standard output about the format of the disk. Use this flag to verify that a device or file system is a DOS disk.

Examples

1. To copy a text file from a DOS diskette to the AIX file system:

   ```
   dosread -a chap1.doc chap1
   ```

   This copies the DOS text file `\CHAP1.DOC` on **/dev/fd0** to the AIX file `chap1` in the current directory.

2. To copy a binary file from a fixed-disk DOS file system to the AIX file system:

   ```
   dosread -D/dev/hd1 /survey/test.dta /u/fran/testdata
   ```

   This copies the DOS data file `\SURVEY\TEST.DTA` on **/dev/hd1** to the AIX file `/u/fran/testdata`.

Files

/dev/fd0 Device name for diskette drive.

Related Information

The following commands: "**dosdel**" on page 345, "**dosdir**" on page 346, and "**doswrite**" on page 350.

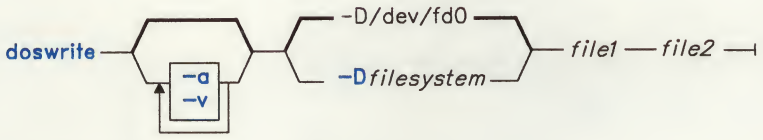
The **pcdos** subroutine in *AIX Operating System Technical Reference*.

doswrite

Purpose

Copies AIX files to DOS files.

Syntax



OL805112

Description

The **doswrite** command copies the specified AIX *file1* to the specified DOS *file2*. If *file2* contains a /, each intervening component must exist as a directory and the last component (the named file), must not exist.

File-naming conventions are those of DOS, with one exception. **doswrite** replaces the \ (backslash) character used to separate components of a DOS path name with the / (slash) because the backslash can have special meaning to AIX. **doswrite** converts lowercase characters in the *file1* name to uppercase before it checks the disk. Because all file names are assumed to be full (not relative) path names, you need not add the initial / (slash).

Notes:

1. Wild card characters (* and ?) are not treated in a special way by this command (although they are by the shell). If, for example, you do not specify a file-name extension, the file name is matched as if you had specified a blank extension.
2. This command must be named **doswrite**.

Flags

-a	Replaces **NL** (new-line characters) with the sequence **CR-LF** (carriage return-line feed). **Ctrl-Z** is added to the output at the end of file.
-D *filesystem*	Specifies the name of the DOS device or file system. The default *device* is **/dev/fd0**. This device must have the DOS-disk format.
-v	Writes information to the standard output about the format of the disk. Use this flag to verify that a device or file system is a DOS disk.

Examples

1. To copy a text file from the AIX file system to a DOS diskette:

   ```
   doswrite  -a  chap1  chap1.doc
   ```

   This copies the AIX file `chap1` in the current directory to the DOS text file `\CHAP1.DOC` on **/dev/fd0**.

2. To copy a binary file from the AIX file system to a fixed-disk DOS file system:

   ```
   doswrite  -D/dev/hd1  /u/fran/testdata  /survey/test.dta
   ```

   This copies the AIX data file `/u/fran/testdata` to the DOS file `\SURVEY\TEST.DTA` on **/dev/hd1**.

Files

/dev/fd0 Device name for diskette drive.

Related Information

The following commands: "**dosdir**" on page 346, "**dosread**" on page 348, and "**dosdel**" on page 345.

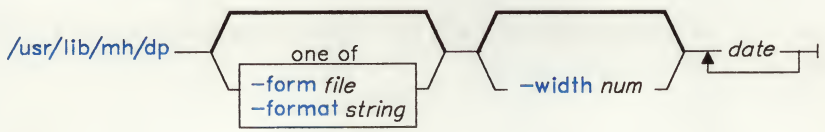
The **pcdos** subroutine in *AIX Operating System Technical Reference.*

dp

Purpose

Parses and reformats dates.

Syntax



AJ2FL227

Description

The **dp** command is used to parse and reformat dates. **dp** is not designed to be run directly by the user; it is designed to be called by other programs. The **dp** command is typically called by its full path name. The **dp** command is part of the MH (Message Handling) package.

The **dp** command parses each string specified as a date and attempts to reformat the string. The default output format for **dp** is the ARPA RFC822 standard. For each string it is unable to parse, **dp** displays an error message.

Flags

-form *file*	Reformats the given dates into the alternate format described in *file*.
-format *string*	Reformats the given dates into the alternate format specified by *string*. The default format string is: `%<(nodate{`*date*`})error:%{`*date*`}%!%(putstr(pretty{`*date*`}))%>`
-help	Displays help information for the command.
-width *num*	Sets the maximum number of columns that **dp** uses to display dates and error messages. The default is the width of the display.

Files

$HOME/.mh_profile	The MH user profile.

Related Information

The MH command "**ap**" on page 53.

The **mh-format** and **mh-profile** files in *AIX Operating System Technical Reference.*

"Overview of the Message Handling Package" in *Managing the AIX Operating System.*

dsipc

Purpose

Installs the Interprocess Communication key mapping in the kernel.

Syntax

OL805461

Description

The **dsipc** command replaces all IPC key mapping currently in the kernel with new mapping from the profile database. The **dsipc** command is usually called, at system startup time, by **/etc/rc.include** to update the Distributed Services kernel. To use **dsipc** command from the command line, you must be a member of the system group or have superuser authority (see "**su**" on page 1026).

Related Information

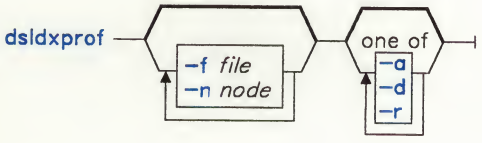
"Using Distributed Services" in *Managing the AIX Operating System.*

dsldxprof

Purpose

Loads translate information into the UID/GID translate profiles.

Syntax



OL805460

Description

The **dsldxprof** command loads translate information from a file into the UID/GID translate profiles. Each line in the file contains a row of translate information in the following format:

Usr/Grp_name U/G Local_id Outbound_id Inbound_id Originating_node

This format is the same as the translate information from the Network Users/Groups Table. You must specify the *U/G*, *Local_id*, and either the *Inbound_id* or *Outbound_id* fields. If you specify the *Inbound_id* field, the *Originating_node* field must also be specified. Data entered after the Originating Nickname/Node ID is treated as a comment and ignored. A - (hyphen) is placed in unused fields as a place holder.

The **dsldxprof** command reads a line of data from the file. If a line begins with a * (asterisk), it is a comment line. Comment lines are ignored and the next line is read. If the line is not a comment line, **dsldxprof** validates the data and loads the data into profiles. Translate rows are rejected due to improper syntax or incorrect values, or they may conflict with translate rows already in the profiles. A translate row is in conflict if there is an existing row in the profiles with a matching *U/G*, *Local_id*, *Inbound_id*, and *Originating_node*, or if there is an existing row in the profiles with a matching *U/G*, *Local_id*, and *Outbound_id*, or both. If there is conflict, you are prompted to replace or reject the conflicting row. Rejected rows are written to standard error along with the information on why they are rejected.

To delete a translate row from the profiles, precede an identical row in the file with ##.

To use **dsldxprof** command, you must be a member of the system group or have superuser authority (see "**su**" on page 1026).

Flags

-a	Places all rows that are in conflict into the profiles without prompting.
-d	Deletes the **pfsuidgid** profile and then recreates it without any entries. This option is handled before the **-f** option if both exist.
-f *filename*	Reads translate information from *filename*.
-n *nodename*	Updates translate profiles on the remote node *nodename*.
-r	Rejects all conflicting rows without prompting.

Files

pfsuidgid	Contains translate information.
.pfsuidgid	Contains indexes defined for **pfsuidgid**.

Related Information

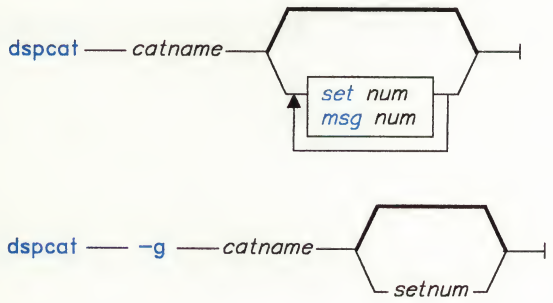
The discussion on distributed services (which includes information on the Network Users/Groups Table) and the discussion on using the **dsldxprof** command in *Managing the AIX Operating System*.

dspcat

Purpose

Displays all or part of a message catalog.

Syntax



OL805482

Description

Use **dspcat** to display a particular message, all of the messages in a set, or all of the messages in a catalog. The syntax for **dspcat** is:

```
$ dspcat catname [set_num] [msg_num]
```

catname specifies a message catalog, *set_num* specifies a set in the catalog, and *msg_num* specifies a particular message in the set. If you include all three parameters, **dspcat** displays a particular message. If you do not include *msg_num* or the *msg_* is in error, all the messages in the set are displayed. If you specify a nonexistent *set_num*, all messages in the catalog are displayed. If you specify only *catname*, all the messages in the catalog are displayed. You must include *set_num* if you include *msg_num*.

Japanese Language Support Information

If Japanese Language Support is installed on your system, this command is not available.

Flags

-g Formats the output so that it can be used as input to **gencat**.

Related Information

The following commands: "**dspmsg**" on page 359, "**gencat**" on page 470, "**mkcatdefs**" on page 651, and " **runcat**" on page 852.

The **catopen**, **catgets**, **catgetmsg**, **catclose**, **NLcatopen**, **NLcatgets**, and **NLgetamsg** files in *AIX Operating System Technical Reference*.

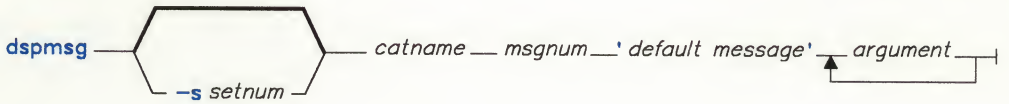
The discussion of **dspcat** in *AIX Operating System Programming Tools and Interfaces*.

dspmsg

Purpose

Displays a selected message from a message catalog.

Syntax



OL805483

Description

dspmsg displays a particular message from a catalog. It allows you to pass up to ten string arguments for substitution into the message if it contains the **printf** conversion specification **%s**, or the **NLprintf** conversion specification **%*n*$s**. The syntax for **dspmsg** is:

```
$ dspmsg catname [-s set_num] msg_num ['default-message' [args]]
```

You must specify the catalog (*catname*) and the message (*msg_num*). The default set number is 1. Specify another set by using the **-s** flag followed by the set number.

If **dspmsg** cannot find the message, the *default-message* is displayed. You must enclose the default message in single quotes if you are using the **%*n*$s** notation for message inserts. If **dspmsg** cannot find the message, and you do not specify a default message, a system-generated error message is displayed.

Follow the default message with up to ten arguments to substitute into the catalog message (or the default message). Missing arguments for conversion specifications are replaced by null strings.

Japanese Language Support Information

If Japanese Language Support is installed on your system, this command is not available.

Related Information

The following commands: "**dspcat**" on page 357, "**gencat**" on page 470, "**mkcatdefs**" on page 651, and " **runcat**" on page 852.

The **catopen**, **catgets**, **catgetamsg**, **catclose**, **NLcatopen NLcatgets**, and **NLgetamsg** files in *AIX Operating System Technical Reference.*

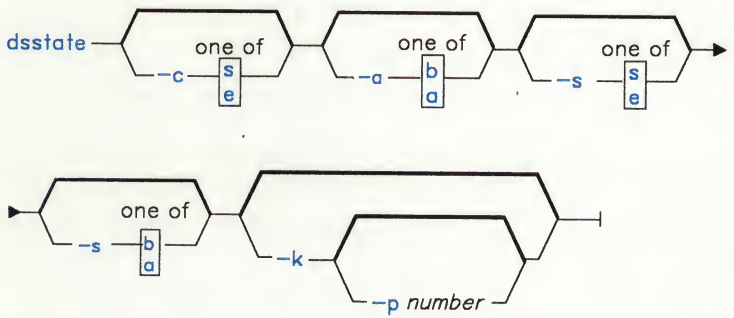
The discussion of **dspmsg** in *AIX Operating System Programming Tools and Interfaces.*

dsstate

Purpose

Sets the state of the Distributed Services kernel logic.

Syntax



OL805462

Description

The **dsstate** command changes the state of the Distributed Services kernel logic, including the number of kernel processes allocated for Distributed Services, whether incoming and outgoing remote requests are allowed, and where temporary storage takes place. Only members of the system group or users operating with superuser authority can use **dsstate** to change the state of the Distributed Services kernel logic (see "**su**" on page 1026). Other users can use **dsstate** with no flags to write to the standard output the current state of the Distributed Services kernel logic.

Flags

-c s Starts client sync, which forces all files for which this node is the client to be written directly to the server, preventing caching (temporary storage) of the file contents at the client. Starting client sync often affects the performance of file operations, and is used primarily for certain system startup and shutdown routines.

-c e Ends client sync and allows some data to be stored at the local node.

-a b Breaks all connections with remote nodes and blocks new requests for remote file services.

-a a Allows requests from this client node for remote file services.

-s s Starts server sync, which forces all files for which this node is the server to be written directly to the server, preventing caching (temporary storage) of the file contents at the client node. Starting server sync often affects the performance of file operations, and is used primarily for certain system startup and shutdown routines.

-s e Ends server sync and allows some data to be stored at the client node.

-s b Blocks all requests for file services from other nodes, including both new requests and requests for files already in use.

-s a Allows this server to accept requests for file services from other nodes.

-k Starts the Distributed Services kernel processes.

-p *number* Sets the number of active Distributed Services kernel processes to *number*. If *number* is greater than the number of kernel processes allocated for Distributed Services, then those that are available are activated. If *number* is 0 or a negative value, the number of kernel processes is not changed.

By adjusting the number of active Distributed Services kernel processes, the rate at which services are provided to remote nodes can be varied. Lowering the number of active Distributed Services kernel processes lowers remote use of this node's processor, leaving more system resources for local use.

Note: The Distributed Services kernel processes must have been started with a **-k** flag on either this **dsstate** command or an earlier **dsstate** command.

Related Information

The **dsstate** system call in *AIX Operating System Technical Reference.*

"Using Distributed Services" in *Managing the AIX Operating System.*

dsxlate

Purpose

Installs Distributed Services UID/GID translate tables into the kernel.

Syntax

OL805463

Description

The **dsxlate** command installs Distributed Services UID/GID translate tables. This command is usually called at system startup by **/etc/rc.ds** to update the kernel. It ensures that the Distributed Services kernel tables reflect the current profiles. All existing Distributed Services kernel information is discarded. To use **dsxlate** command, you must be a member of the system group or have superuser authority (see "**su**" on page 1026).

Related Information

The following commands: "**ipctable**" on page 544, "**ndtable**" on page 685, and "**ugtable**" on page 1109.

The **loadtbl** system call in *AIX Operating System Technical Reference.*

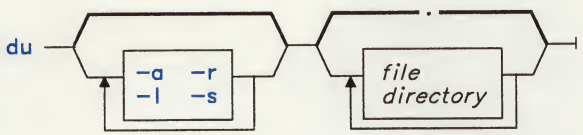
"Using Distributed Services" in *Managing the AIX Operating System.*

du

Purpose

Summarizes disk usage.

Syntax



OL805113

Description

The **du** command gives the number of blocks in all files and (recursively), directories within each specified *directory*. By specifying the **-a** flag, you can also have **du** report the number of blocks in individual files. The block count includes the indirect blocks of each file and is in units of 512 bytes, independent of the cluster size used by the system. If you provide no *file* or *directory* name, **du** uses the current directory.

Notes:

1. If you do not specify the **-a** flag, **du** does not report on any *file*s.
2. If there are too many distinct linked files, **du** counts the excess files more than once.
3. Block counts are based only on file size; therefore, unallocated blocks are not accounted for in the block counts reported.

Flags

-a Displays disk use for each file.

-l Allocates blocks in files with multiple links evenly among the links. By default, a file with two or more links is counted only once.

-r Indicates inaccessible files and directories.

-s Displays only the grand total (for each of the specified files or directories given).

Examples

1. To summarize the disk usage of a directory tree and each of its subtrees:

   ```
   du  /u/fran
   ```

   For `/u/fran` and each of its subdirectories, this displays the number of disk blocks that the files in the tree beneath it contain.

2. To display the disk usage of each file:

   ```
   du  -a  /u/fran
   ```

   This displays the number of disk blocks contained in each file and subdirectory of `/u/fran`. The number beside a directory is the disk usage of that directory tree. The number beside a regular file is the disk usage of that file alone.

3. To display only the total disk usage of a directory tree:

   ```
   du  -rs  /u/fran
   ```

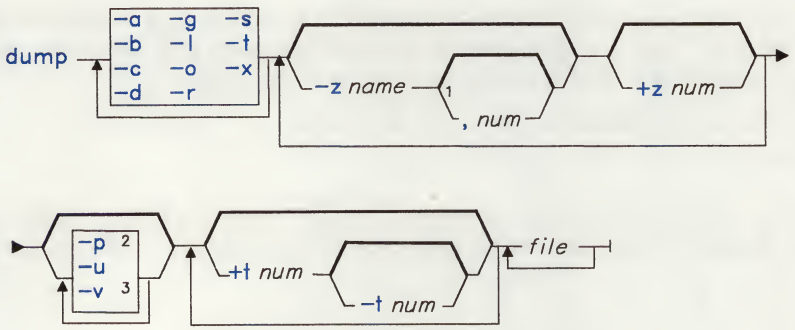
   This displays only the sum total disk usage of `/u/fran` and the files it contains (`-s`). The `-r` flag tells **du** to display an error message if it cannot read a file or directory.

dump

Purpose

Dumps selected parts of an object file.

Syntax



[1] Do not put a space between these items.

[2] Use -p only with -a, or -o .

[3] Do not use -v with -s or -o .

OL805404

Description

The **dump** command dumps selected parts of the specified *file*. **dump** accepts object files, archive object files, and executable files (with the **-x** flag). It writes information in character, hexadecimal, octal, or decimal representation, as appropriate to format the information in a meaningful way.

Flags

You must use at least one of the following flags:

-a Dumps the archive header of each member of each specified archive.

-b Dumps the shared library key.

-c	Dumps the string table.
-d	Dumps the contents of the data section.
-g	Dumps the global symbols in the archive symbol table.
-l	Dumps line number information.
-o	Dumps each optional header.
-r	Dumps relocation information.
-s	Dumps the contents of the object file section.
-t	Dumps symbol table entries.
-x	Dumps the object module extended header from executable files. The extended header contains the table of shared libraries that the program uses.

The following optional flags are also available:

-p	Does not print the headers.
-t*num*	Dumps only the index symbol table entry specified with *num*. Use **-t** with the **+t** flag to specify a range of symbol table entries.
+t*num*	Dumps the symbol table entry in the range that ends with *num*. The range starts at either the first symbol table entry or at the entry specified by **-t**.
-u	Underlines the name of the *file*.
-v	Dumps the information in symbolic representation rather the numeric. You can use this with any of the above flags except **-s** or **-o**.
-z*name*[,*num*]	Dumps line number entries for *name* function or a range of line number entries that starts at the specified number. You can use a blank to replace the comma that separates *name* and *num* if the entire argument is quoted.
+z*num*	Dumps all line numbers up to *num*.

Related Information

The following commands: "**ar**" on page 55, "**nm**" on page 705, "**shlib**" on page 939, and "**size**" on page 949.

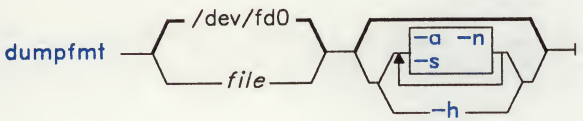
The **a.out** and **ar** files in *AIX Operating System Technical Reference*.

dumpfmt

Purpose

Formats the VRM dump file.

Syntax



OL805109

Description

The **dumpfmt** command formats a file containing VRM dump structures. If you do not specify a *file* name, the system reads data from **/dev/fd0**.

By default, **dumpfmt** is an interactive utility program. To see the list of commands available for selecting a specific structure to format, enter a ? (question mark). To quit, enter **q**.

Flags

-a Batches the output and formats the entire diskette.

-h Includes a Dump Data Header. This header contains general information about data on the dump diskette: the module name of the component, the data address of the module containing the component, and the offset address within the module of the component.

-n Does not display a prompt when the screen fills with data during interactive output.

-s Limits the output of each structure to a maximum size of 32 bytes.

Related Information

The discussion of **dumpfmt** in *AIX Operating System Programming Tools and Interfaces.*